4-2020

Two can play that game: An adversarial evaluation of a cyber-alert inspection system

Ankit SHAH

Arunesh SINHA
Singapore Management University, aruneshs@smu.edu.sg

Rajesh GANESAN

Sushil JAJODIA

Hasan CAM

Two Can Play That Game: An Adversarial Evaluation of a Cyber-Alert Inspection System

ANKIT SHAH, University of South Florida
ARUNESH SINHA, Singapore Management University, Singapore
RAJESH GANESAN and SUSHIL JAJODIA, George Mason University
HASAN CAM, Army Research Lab

Cyber-security is an important societal concern. Cyber-attacks have increased in numbers as well as in the extent of damage caused in every attack. Large organizations operate a Cyber Security Operation Center (CSOC), which forms the first line of cyber-defense. The inspection of cyber-alerts is a critical part of CSOC operations (defender or blue team). Recent work proposed a reinforcement learning (RL) based approach for the defender's decision-making to prevent the cyber-alert queue length from growing large and overwhelming the defender. In this article, we perform a red team (adversarial) evaluation of this approach. With the recent attacks on learning-based decision-making systems, it is even more important to test the limits of the defender's RL approach. Toward that end, we learn several adversarial alert generation policies and the *best response* against them for various defender's inspection policy. Surprisingly, we find the defender's policies to be quite robust to the best response of the attacker. In order to explain this observation, we extend the earlier defender's RL model to a game model with adversarial RL, and show that there exist defender policies that can be robust against any adversarial policy. We also derive a competitive baseline from the game theory model and compare it to the defender's RL approach. However, when we go further to exploit the assumptions made in the Markov Decision Process (MDP) in the defender's RL model, we discover an attacker policy that overwhelms the defender. We use a *double oracle* like approach to retrain the defender with episodes from this discovered attacker policy. This made the defender robust to the discovered attacker policy and no further harmful attacker policies were discovered. Overall, the adversarial RL and double oracle approach in RL are general techniques that are applicable to other RL usage in adversarial environments.

CCS Concepts: • **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

Additional Key Words and Phrases: Cyber-security operations center, adversarial reinforcement learning, game theory

ACM Reference format:
Ankit Shah, Arunesh Sinha, Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. 2020. Two Can Play That Game: An Adversarial Evaluation of a Cyber-Alert Inspection System. *ACM Trans. Intell. Syst. Technol.* 11, 3, Article 32 (March 2020), 20 pages.
https://doi.org/10.1145/3377554

A. Shah and A. Sinha contributed equally to this research.
This work was supported in part by the Army Research Office under MURI grant W911NF-13-1-0421.
Authors' addresses: A. Shah, University of South Florida, 4202 E. Fowler Ave., ENG-030, Tampa, FL 33620-5530; email: ankitshah@usf.edu; A. Sinha, Singapore Management University, 80 Stamford Road, Singapore 178902; email: aruneshs@smu.edu.sg; R. Ganesan and S. Jajodia, Center for Secure Information Systems, George Mason University, 4400 University Dr., MS 5B5, Fairfax, VA 22030-4422; emails: {rganesan, jajodia}@gmu.edu; H. Cam, Army Research Laboratory, 2800 Powder Mill Road, Adelphi, MD 20783-1138; email: hasan.cam.civ@mail.mil.

2157-6904/2020/03-ART32 $15.00
https://doi.org/10.1145/3377554

1 INTRODUCTION

In this era of truly pervasive computing, cyber-security has emerged as a major concern. Cyber-attacks have crippled large hospitals [1] and resulted in stolen sensitive information from large companies as well as defense agencies [40]. Most large organizations, including defense agencies, operate a Cyber Security Operation Center (CSOC). A CSOC is a team of specialized analysts, engineers, and responders responsible for maintaining and improving cyber-security. A critical task in the CSOC is to inspect cyber-alerts generated by various sensing software such as SNORT or tools such as ArcSight. Given the high false positive rates of cyber-alerts [25], it is important to screen the alerts effectively to identify any real attack signal from these alerts. It is also required to maintain the queue length of alerts within acceptable limits.

The above inspection operation in a CSOC is partitioned into levels. The first level of inspection is running some standard checks in order to determine the severity of the alert as well as the need for a secondary detailed inspection. A recent work [36], in collaboration with a real-world CSOC, developed a reinforcement learning (RL) based approach (the defender's RL is henceforth called CSOC-RL) to manage the resources (analyst time) needed for the first level of inspection of cyber-alerts such that the queue length of alerts remained within acceptable limits. The model in the CSOC-RL work is quite detailed to match actual operations in the real-world CSOC, but, the adversary was modeled as a stochastic agent using a probabilistic distribution for generating the alerts. Distinct from CSOC-RL, another work [35] uses a game theoretic approach for cyber-alerts inspection, but, the model used simplifying assumptions such as a fixed number of analysts and a fixed number of alerts arriving every hour, as well as a single shot interaction.

The motivation for this research stems from the fact that the above CSOC-RL approach has not been tested against an intelligent adversary who could also learn several attack responses (using another RL) in order to overwhelm the defender and increase the queue length of alerts. Further, with the recent attacks on learning-based decision-making systems, it is even more important to test the limits of this RL approach. In this work, we perform a red team (attacker) evaluation of the CSOC-RL approach for the alerts that require detailed inspection by analysts, using both empirical and theoretical techniques. The objective of the research is to test the efficacy of the CSOC-RL approach against an intelligent adversary. The research work presented here is different from the CSOC-RL approach because the attacker is no longer a probability distribution, and instead is another RL agent with the goal to increase the queue length of the alerts for the defender. To this end, the model presented here is a game model with the defender having a fixed total quantity of additional resources (analyst time) to handle a fixed additional total number of alerts generated by the attacker. However, what is not known is the timing and quantity of alerts that will be generated (adversary's action) and the timing and quantity of resources that the defender must deploy (defender's action) over a two-week work cycle of the CSOC such that the queue length of alerts remains within acceptable limits. It should be noted that the research tests the defender's CSOC-RL approach for *additional* alerts. This means that the CSOC is already operating for a certain base level of alert generation, which is deemed as the normal operating condition of the CSOC. The normal operation has an acceptable queue length of alerts, which establishes the normal level of operational effectiveness (LOE), according to Shah et al. [37]. The additional alerts would increase the queue length (decrease the LOE of the CSOC), which in turn prompts the defender to deploy additional resources.

Our *first contribution* is to pose the attack policy learning problem of an attacker as a RL problem itself, given a deployed defender policy that was also learned using RL but against a stochastic adversarial agent. The CSOC-RL model for decision-making is based on an underlying queuing process of cyber-alerts (explained in Section 3). The defender has a base number of analysts and a

given budget of *additional* analysts for the alert inspection. The action of the defender is to allocate (or not) additional analysts every hour depending on the state of the system, where the state is roughly the amount of backlog of alerts. The defender aims to keep the backlog of alerts below a threshold provided by the CSOC. We model the attacker analogously as choosing to send *extra* alerts over and above the base stochastic arrival of alerts based on the underlying queuing process, given a fixed total budget of additional alerts. For the first set of experiments, the attacker's aim is to push the backlog of alerts above the given threshold, which results in no or less rigorous checking of alerts leading to increased chance of actual attack going undetected. These attacks that overwhelm the system have been observed in practice [9]. Surprisingly, our experiments reveal that the attacker fails to overwhelm the system (defined quantitatively in Section 5) when its budget of additional alerts is the same as that of the defender's budget of additional inspections, even though the defender policy was learned against a stochastic adversary.

In order to understand the failure of the above attacker policy, we formulate a zero-sum game version of the CSOC-RL model as our *second contribution*. This CSOC-GAME model is a partially observable stochastic game but where the total reward is the maximum of the reward in every time step, as opposed to the standard discounted sum of rewards. Building on recent results in stochastic games [17], we show that the minimax theorem holds for this non-standard game, which reveals that there exists a defender's policy that is robust against *any* attacker policy. The experimentally observed failure of the attacker's best response to the learned defender policy implies that the defender is learning a policy close to its minimax policy (discussed later in Section 4.2). Digging further into queuing theory, we derive simple rule-based defender policies that guarantee certain minimum reward for the defender. We find experimentally that these rule-based policies, while simple and interpretable, are inferior in performance to the RL learned defender policy.

Our *third contribution* is a successful attacker policy that works by exploiting a modeling assumption in the CSOC-RL model and a defense against this attack. In the CSOC-RL model, operational considerations restricted the number of additional resources allocated by the defender to be in discrete chunks of fixed size. Importantly, the CSOC-RL work also assumed that the adversary sends alerts in discrete chunks of exactly the same size as the defender's inspection chunks, which we exploit in our attack. By relaxing the assumption on the attacker, we find an attacker policy that exhausts the defender budget of additional inspections using a small number of additional alerts that are sent several times, which then allows the attacker to overwhelm the system arbitrarily. It was observed that the attacker exploited the fixed size of inspection chunks to elicit a disproportionate response to few additional alerts. Inspired by the double oracle method from game theory, we retrain the defender using additional episodes from the discovered attacker policy and find that the defender's policy becomes robust to the discovered policy. The relearned defender policy now exhibits more patience and allows the backlog to build up more before allocating additional inspections compared to the prior policy. In the next double oracle iteration, no harmful attacker policies were discovered, thus, providing evidence that the relearned defender policy is robust to any adversarial generation policy.

The article is organized as follows: Section 2 sets our work in relation to other work; Section 3 provides the necessary details on background and prior work, and also introduces the notation for this article, Section 4 describes our red team evaluation methodology including both attacks and defenses, Section 5 presents our experiments showing the results of attacks and defenses, and, finally, we conclude in Section 6.

2 RELATED WORK

A CSOC protects an organization by employing cyber-security analysts who investigate suspicious activities that are flagged [5, 12, 32] by automated filters (intrusion detection systems and secure

information and event management systems [6]) in the form of alerts. Bi-weekly (14-day) schedules of analysts are created to maximize alert investigations in each of the work-shifts [16]. The initial alert analysis (first level of investigation) is a fast decision-making process by the analysts [13]. The alerts are categorized as innocuous or significant alerts at the end of the first level of inspection. The significant alerts are then passed on to the secondary level of investigation, which may take hours or days to finish analysis. There are adverse events such as adversarial attacks that impact a CSOC by causing delays in alert investigations. If the situation is not rectified soon, it may prove costly for the organization that is monitored by the CSOC due to a delay in the timely detection of an attack.

Shah et al. [37] propose a novel metric to quantify delays in alert investigations at a CSOC. The performance of the CSOC is quantified using this metric, average total time for alert investigation (AvgTTA), which is the average of the sum of waiting time in the queue and the investigation time of all the alerts investigated in each time period (for example, hourly). The performance of the CSOC, measured in terms of the AvgTTA metric values, is continuously monitored using a color-coded representation. In Reference [38], the authors propose a tradeoff model to establish a CSOC with the right types and numbers of full-time analysts such that an optimal value of AvgTTA is maintained, given a limited budget to hire the full-time analysts and an estimated alert generation rate. The proposed model establishes the steady state conditions between the estimated demand for alert investigation and the alert analysis capacity at a CSOC.

The authors in Reference [36] propose a decision-making framework using RL to control this metric under stochastic conditions that increase the delays in alert investigations beyond the steady state (baseline) value, by dynamically allocating a limited number of additional resources that are available at a CSOC in a 14-day time period. The CSOC-RL model is tested against stochastic events with a Poisson arrival process and the results indicate that the RL approach works better than the rule-based strategies employed by the CSOC operators against random adverse events.

Game theoretic inspection or auditing has appeared in many papers [7, 8, 42]; however, all these works have a single shot interaction model and are not focused on cyber-alert inspection. As stated earlier, a game-based work [35] on cyber-alert inspection is modeled as a single shot interaction model, which has other stringent assumptions such as a fixed number of analysts and alerts generated per hour. Another recent game theoretic work [29] uses a zero-sum Markov game model for cyber-alert inspection, but the model assumes complete information for both the players and, hence, solves the game using standard minimax value iteration. Game theory has also been used for other problems in cyber-security such as deception [34], attack graphs [14], man-in-the-middle attacks [23], and spear fishing [22, 43], which are quite different from the cyber-security problem presented in this article. Also, scalability is still an issue in solving partially observable stochastic games [19].

Recent work on adversarial attacks on learning techniques have found attacks on deep RL systems [4, 20, 24]. A recent work [30] looks at the adversarial problem as a zero-sum stochastic game with complete information for both players and proposes a best response dynamics approach to solve the problem; that is, alternatively, each player plays its best response fixing the other player's last policy. However, there is no guarantee of best response dynamics converging to an equilibrium, even in zero-sum games (for example, matching pennies). Our problem is harder due to the partial observation of current state. We utilize a different technique where the successful attack of the attacker is incorporated as episodes in the RL training of the other player, which is inspired by the double oracle technique in game theory. Double oracle techniques have been used in game settings [21] and adversarial settings also [41], but is computationally very costly. Our approach of including only the discovered and valid attacks as training episodes for the defender is a simplification that allows for scalability. We also show that robust performance does not just mean reaching

the equilibrium but also depends on the value of the equilibrium, which in our case relates to the resources (budget) of players.

Finally, queueing processes are well established as a natural model for arrivals and service [10, 11]. Some work extends the classic queueing processes to deal with multiple rational customers who can opt-out of joining a queue [2] or strategic selection of scheduling criteria such as first come, first served (FCFS) [3].

The work presented in this article differs from the work in the literature in the following ways:

(1) The CSOC-RL model proposed in Reference [36] is tested against stochastic events. However, in order to deploy this model, it must be tested against a strategic adversary. In this work, we propose an adversary who can fully observe and interact with the CSOC-RL model and uses an RL framework to learn the best policy of actions to test the robustness of the CSOC RL model. Such an interaction for evaluation would be an important addition to the real-world applicability of the CSOC-RL model.
(2) The game theoretic approach presented in this work accounts for the realistic scenario of incomplete information and signals, and explains theoretically and empirically the subtle relation between the budget of the players. Further, our game has a non-standard long-term utility (see model later).
(3) The model presented in this work has an adversarial interaction of two players on top of the queuing process, which, as far as we know, has not been addressed in the queuing theory literature.

3 BACKGROUND, PRIOR WORK, AND NOTATION

In this section, we provide a brief summary of the prior CSOC-RL work as well as a brief background about the queuing process used in that work. The arrival of alerts was modeled as a Poisson process with the nominal rate of $\lambda_0 = 1{,}919$ alerts/hour, which was chosen based on the inputs obtained from the CSOC operators. The adversary was modeled as a fixed stochastic adversary that changed the actual alert arrival rate λ ($\lambda \geq \lambda_0$) according to an unknown stochastic distribution. This CSOC-RL work modeled the first level of inspection in a CSOC. The first level of inspection is a fast inspection that decides the severity of the alert and whether a follow-up second detailed inspection is required. Given the almost same steps for all alerts in this first level of inspection, the inspection time for every alert is the same. Based on the arrival nominal rate λ_0 and the time to service an alert, a nominal number of analysts were chosen so that the aggregate *nominal* service rate of alert was $\mu_0 = 1{,}920$ alerts/hour. However, the actual service rate μ varied stochastically with an unknown distribution (always $\leq \mu_0$) due to factors such as analyst absenteeism, failure of sensors, etc. We skip the details of analyst scheduling in Reference [36], as that is not required for this exposition.

Background on queuing theory: The above model with fixed rates ($\mu = \mu_0, \lambda = \lambda_0$) is exactly an M/D/1/FCFS queue (this notation is the standard Kendall notation from queuing theory). M stands for Poisson arrival, D for deterministic service time, 1 for the number of servers (here, all analyst are clubbed together as one server), and FCFS means that the alerts are inspected on a first-come-first-serve basis. The M/D/1/FCFS queue has been studied a lot and can be viewed as a discrete time Markov chain with infinite state space $\{0, 1, \ldots\}$ that represents the queue length. The transition probability of this Markov chain depends on λ_0, μ_0. Let A_t, Z_t be the random variable that denotes the number of arrivals and number of alerts serviced in the t^{th} hour. With fixed nominal rate, $P(Z_t = \mu_0) = 1$ (deterministic service) and $P(A_t = n) = \frac{\lambda_0^n \exp(-\lambda_0)}{n!}$ (Poisson distribution). Given queue length b_{t-1} at time $t-1$, the transition probability can be expressed as a function h of λ_0, μ_0 as follows $P(b_t \mid b_{t-1}) = P(A_t - Z_t = b_t - b_{t-1}) = h(\lambda_0, \mu_0, b_t - b_{t-1})$. An

important aspect of M/D/1/FCFS queue is the ratio $\rho_0 = \frac{\lambda_0}{\mu_0}$. The queue is stable with a finite expected queue length only when $\rho_0 < 1$. With the given nominal rates, we have $\rho_0 = 0.999479$.

MDP of the CSOC-RL problem: However, note that with varying λ, μ (with unknown distribution), the CSOC-RL model is not exactly a M/D/1/FCFS queue. Yet, it can still be described by a Markov decision process with the unknown transition probability as $P(b_t \mid b_{t-1}) = P(A_t - S_t = b_t - b_{t-1}) = \int h(\lambda, \mu, b_t - b_{t-1}) p(\lambda, \mu)\, d\lambda\, d\mu$, where $p(\lambda, \mu)$ is the joint (unknown) density of the randomly varying λ, μ. As the transition probability is unknown, the CSOC-RL work modeled the defender's problem as an RL problem. Note that as $\lambda \geq \lambda_0$ and $\mu \leq \mu_0$, the instantaneous $\rho = \frac{\lambda}{\mu}$ can be > 1. Thus, the defender is provided with a total of $X = 28{,}800$ additional inspections to be used over N timesteps, and the λ, μ are so controlled so that the additional alerts (counted as total additional number of alerts due to higher λ and fewer inspections due to lower μ) is also not more than X. The defender-adversary interaction was modeled over $N = 336$ hours (two weeks) as the staffing changes every two weeks. In every hour, the defender could call up to 10 extra analysts, which translated to at most $E = 2{,}400$ additional inspections per hour. Also, an analyst has to be allocated for a minimum of 15 minutes, which translated to a discrete allocation of additional inspections in chunks of $M = 60$ alerts. Formally, the RL problem was modeled with

- *State*: $s \in S$ is a tuple $s = \langle b, n, x \rangle$, where b is the backlog of alerts at the end of time interval t; n is the number of time intervals remaining; and x is the resources remaining for the defender. The initial budget for the defender is X. The initial value of n is the time horizon N.
- *Action*: The defender's action is to allocate d ($d \leq x$) additional inspection resources at the start of the time interval. d is a multiple of M and an integer between 0 and E.
- *Transition*: In the next state, n decreases by one, x decreases by d, and the next b' is given by $P(b' \mid s, d) = P(A_t - Z_t = d + b' - b) = \int h(\lambda, \mu, d + b' - b) p(\lambda, \mu)\, d\lambda\, d\mu$, where A_t, Z_t are defined above.
- *Rewards*: The immediate reward has two terms. The first is due to the cost incurred by the defender from the backlog after allocating additional inspections given by $C(s, d) = f(b - d)$, where function f normalizes the cost to lie between $[0, 1]$ with the value increasing (not strictly) with its argument. The second term is $q(x, n)$, which incentivizes preserving additional resources per time remaining, i.e., q increases with increasing ratio x/n and normalized to lie in $[0, 1]$. Thus, the immediate reward is $-f(b - d) + q(x, n)$. The term q serves as a reward shaping term.[1]

Details of the RL training, the function q, and the size of the problem are provided in the CSOC-RL paper.

Measuring performance: While the RL model described above includes a reward for preserving budget, the effective reward for the defender is only from backlog. The purpose of the term q for preserving budget was to converge quickly to learn to not exhaust all budget. Thus, as presented in the CSOC-RL work, we also show rewards only for the backlog term. In more details, the function $f(x)$ is a piecewise linear function defined as

$$f(x) = \begin{cases} 0 & \text{for } x \leq L \\ \frac{x-L}{U-L} & \text{for } L < x < U \\ 1 & \text{for } U \leq x \end{cases}$$

[1] A reward shaping term in reinforcement learning encourages faster convergence of the learning [18, 28]. q does not directly contribute to the defender's cost.

In the prior CSOC-RL work, $L = 1{,}175$ ($U = 4{,}350$) alerts corresponds to 1 hour (4 hours) worth of average wait time to inspect an alert (AvgTTA). The results in the CSOC-RL paper show the backlog in terms of AvgTTA, which we elaborate on further in the experiments section. The f function was designed in consultation with experts from the real-world CSOC, and f is a simple linear function normalized to lie in [0,1] between the acceptable one hour backlog and the unacceptable 4 hours backlog. From our conversation with real-world CSOC operators, the defender's aim is to keep the maximum AvgTTA over the time horizon N as low as possible, with anything over the 4-hour threshold being unacceptable.

4 ADVERSARIAL EVALUATION METHODOLOGY

In this section, we present our approach to the red team evaluation. The approach has three distinct parts that are presented in the following sections.

4.1 Adversarial RL

Recall that the attacker in the CSOC-RL work was a fixed stochastic agent that changed the actual alert arrival rate λ ($\lambda \geq \lambda_0$) according to an unknown stochastic distribution. We make the attacker truly adversarial by allowing him to control the number of alerts to send in every timestep, subject to a total budget constraint Y. The attacker's optimal attack problem can be set up as an RL problem itself. The MDP of the adversarial RL problem is described as:

- *State space*: $s \in S$ is a tuple $s = \langle b, n, x, y \rangle$, where b, n, x are the same as for the defender MDP. y is the additional alerts remaining for the attacker. The initial budget for the attacker is Y.
- *Action space*: The adversary action is to send a ($a \leq y$) additional number of alerts at the start of the time interval. a is a multiple of M and an integer between 0 and Y.
- *Transition*: In the next state, n decreases by one, x decreases by d (d given by the fixed and known defender policy), y decreases by a, and the next b' is given by $P(b' \mid s, a) = P(A_t - Z_t - d + a = b' - b) = \int h(\lambda_0, \mu, b' - b + d - a) p(\mu)\, d\mu$.
- *Rewards*: The immediate reward has two terms. The first is exactly the cost incurred by the defender $C(s, d) = f(b - d)$. The second term is $q(y, n)$, which incentivizes preserving additional alerts per time remaining, i.e., q increases with increasing ratio y/n and normalized to lie in $[0, 1]$. Thus, the immediate reward is $f(b - d) + q(y, n)$. Similar to the defender RL, the term q is a reward shaping term that encourages faster convergence of the learning.

Few points to note about the adversary model are: (1) the adversary is very powerful as it has complete knowledge of the backlog b and the defender state x; (2) the base alert arrival rate λ is fixed to λ_0, since the additional alerts are all controlled by the attacker, but the μ still varies randomly; (3) the adversary has no hard bound on the number of additional alerts per hour (like E for the defender), but the q function acts as a soft bound for the number of additional alerts/hour; and (4) the adversary model assumed here allows attacks *only* by sending additional alerts over and above the base number of stochastically generated alerts.

As we show later in experiments, the defender policy learned from the prior RL approach is robust to this attack when $Y \leq X$. This is surprising, as learning methods, including RL, have been shown to be vulnerable to attacks. In order to understand and explain this robustness, we analyze the defender-adversary interaction in a game theory model in the next section.

4.2 Game Theoretic Model

We start by presenting a unified model of a two-player zero-sum repeated game formulated in a recent book by the authors of Reference [27]. Our CSOC-GAME will be presented as an instance of such games. A zero-sum unified repeated game with signals $(S, I, J, C, D, \pi, q, g)$ is defined by a

set of states S, two finite sets of actions I (for player 1) and J (for player 2), two sets of signals C (for player 1) and D (for player 2), an initial distribution $\pi \in \Delta(S \times C \times D)$, and a transition function q from $S \times I \times J$ to $\Delta(S \times C \times D)$, where $\Delta(\mathcal{S})$ denotes the set of probability distributions on a given set $\mathcal{S}$. The reward function for player 1 is given by function $g : S \times I \times J \to [0, 1]$. The reward for player 2 is $-g$. At each stage, t players choose actions i_t and j_t, and a triple $(s_{t+1}, c_{t+1}, d_{t+1})$ is drawn according to $q(s_t, i_t, j_t)$ (for $t = 1$, the draw is according to π), where s_t is the current state, inducing the signals c_{t+1}, d_{t+1} received by the players and the state s_{t+1} at the next stage. Player 1's history at stage t is $c_1, i_1, \ldots, c_{t-1}, i_{t-1}, c_t$; similarly, player 2's history is $d_1, j_1, \ldots, d_{t-1}, j_{t-1}, d_t$. The history of the game is $c_1, d_1, i_1, j_1, \ldots, c_{t-1}, d_{t-1}, i_{t-1}, j_{t-1}, c_t, d_t$. Given perfect recall, a behavioral strategy for player 1 is a sequence $\sigma = (\sigma_t)_{t \geq 1}$, where σ_n, the strategy at stage t, is a mapping from possible histories to $\Delta(I)$, with the interpretation that $\sigma_t(h^1)$ is the mixed action used by player 1 after its history h^1. Similarly, a behavioral strategy for player 2 is a sequence $\tau = (\tau_t)_{t \geq 1}$. This game model is very general and a suitable choice of signal space can model repeated and stochastic games with perfect or imperfect information. Given the above model, an evaluation function maps infinite game histories to total reward. In a typical repeated game, this evaluation function is a discounted sum of the per stage rewards given by g.

The defender adversary interaction is a game on top of an underlying stochastic process of arrival and processing of cyber-alerts, with varying rates of arrival λ and service μ (stated in Section 3). In order to model the defender attacker interaction as a unified repeated game model, we remove two heuristic choices made in the RL models. We first remove the term q in the defender reward, as that term was used only for faster convergence. Next, we remove term q in the adversary reward, and instead place a hard bound E on the number of additional alerts per hour. The game model is as follows:

- Player 1 is the attacker and player 2 is the defender.
- *State space*: Each state $s \in S$ is a tuple $s = \langle b, n, x, y \rangle$ with the exact same specification as the MDP of the adversary RL.
- *Action spaces* I and J: The adversary action is to send a an additional number of alerts over at the end of the time interval. The defender action is to allocate d additional inspection resources at the start of the time interval. a and d are both multiples of M, and both are integers between 0 and E.[2]
- *State transition*: Recall that A_t is the random number of alerts arriving due to the underlying Poisson process with random rate λ in the time interval t. Similarly, Z_t is the number of alerts processed at random service rate μ. Given $s = \langle b, n, x, y \rangle$ with $n > 0$ and actions d and a, the resultant state $s' = \langle b', n', x', y' \rangle$ satisfies $n' = n - 1$ if $n > 0$, $x' = \max(0, x - d)$, $y' = \max(0, y - a)$, and $b' = b - \min(d, x) + \min(a, y) + A_t - Z_t$. Let E denote $b' - b + \min(d, x) - \min(a, y)$. It can be seen that $P(s' \mid s, d, a) = P(b' \mid s, d, a) = P(A_t - Z_t = E) = \int h(\lambda_0, \mu, E) p(\mu)\, d\mu$. States with $n = 0$ are sink states.
- *Time horizon*: While the game is of finite horizon N, we model it as infinite horizon by fixing the rewards for both players in any state with $n = 0$ to 0 (see the next item).
- *Rewards*: The cost incurred by the defender is the backlog after allocating additional inspections, i.e., $C(s, d) = f(b - d)$. The immediate reward for players is:
  - Attacker: $g(s, d, a) = C(s, d)$ when $n > 0$, else 0
  - Defender: $-g(s, d, a) = -C(s, d)$ when $n > 0$, else 0

  The game is clearly zero-sum.

[2]The game model is well-defined when the action space does not change over time. The budget constraint on action is indirectly imposed in the state transition using the min functions.

— *Signal space* C and D: Both players observe separate signals after their actions. The attacker has complete observation; thus, the attacker's signal after the current timestep is the next state s' and d, a. The defender does not observe the attacker's action a and the y part of the state. However, the defender receives signals for these, which is the backlog b'. Thus, the signal for the defender is b', n', x', d. The signal probability for the defender is defined exactly like the state transition probability $P(b' \mid s, d, a)$.
— *History of signals*: At any time point T the history observed by the adversary is a history s_t, d_t, a_t for all $t \leq T$. For the defender, the observed history is b_t, n_t, x_t, d_t for all $t \leq T$. A play of the game till time T is defined as s_t, d_t, a_t for all $t \leq T$ (this is the same as the observed history of the attacker as an adversary observes all the past). All infinite plays of the game are denoted by the set H_∞.
— *Strategy*: As defined earlier, strategies are functions of observed player histories to mixed actions. For the attacker, strategy σ is a function from all past states and both players' action. For the defender, strategy τ is a function of all past b, n, x, and d.
— *Evaluation function*: In contrast to the standard stochastic game, the defender only cares about the maximum length of the queue of alerts over the time intervals. Thus, the long term reward for the defender is the inf of the rewards over all time intervals (note that the defender stage reward is negation of cost; hence, inf captures the worst backlog over time). For the attacker, due to the antagonistic nature of interaction, the long-term reward is then $sup(h) = \sup_{t \geq 1} g(s_t, d_t, a_t)$ for $h \in H_\infty$.

Fortunately, even for such a complex game, a minimax theorem holds for sup evaluation [17] (which is not true for many other evaluation functions), thus,

$$\sup_{\sigma} \inf_{\tau} E_{\sigma,\tau}(sup(h)) = \inf_{\tau} \sup_{\sigma} E_{\sigma,\tau}(sup(h)) = V$$

The above result can also be interpreted from the defender's perspective that there is a strategy τ^* that can achieve value $-V$ irrespective of the strategy used by the adversary. However, the magnitude of the value V is important (e.g., $V = 1$ is not very good for the defender as it provides the lowest utility possible -1). We show below that the budgets of the players decide what V will be achieved for the special case of fixed service rate. Fixed service rate is a mild assumption as analyst absenteeism is rare and accounted for by a buffer of additional analyst time. Further, as defined, these game strategies can be non-Markovian; that is, these strategies depend on the history of backlogs. Reinforcement learning learns policies, which, by definition, are Markov strategies that only depend on the last state. However, our result below constructs a simple Markov strategy that achieves a minimum value for the defender showing that the space of Markov strategies also contains policies that guarantee high minimum value for the defender.

Theorem 1. *Given fixed $\mu = \mu_0$, the defender can guarantee himself a long term reward of (a) at most $-[1 - \exp(-\epsilon^2\lambda/2T)] * f(R - \mu * T + (T - \epsilon) * \lambda)$, if $Y - X \geq R$, where T, ϵ ($T < N$) can be chosen to maximize the loss and (b) at least $(1 + f(B))(1 - 1/B)^{N\mu/B} - 1$, if $Y \leq X$. Further, the guarantee of case (b) is provided by a simple rule-based policy: (S1) whenever the backlog exceeds B by x, allocate x additional inspections.*

Proof. First, the case when $Y = X + R$; the case when $Y > X + R$ can be analyzed in exactly the same way as below by assuming that the adversary uses only $X + R$ alerts. Consider the attacker sending $B = Y/T$ alerts every hour from the start. In T hours, the attackers send Y additional alerts. In T hours, the number of arrivals S from the Poisson process form a Poisson distribution with rate $T\lambda$. The following concentration inequality is known for Poisson distribution [31]

$$P(S \leq T\lambda - \epsilon\lambda) \leq \exp\left(-\frac{\epsilon^2\lambda^2}{2T\lambda}\right) = \exp\left(-\frac{\epsilon^2\lambda}{2T}\right)$$

Thus, with probability $\geq 1 - \exp(-\frac{\epsilon^2\lambda}{2T})$, the number of normal alerts arrivals in T hours is more than $(T-\epsilon)\lambda$. The number that can be served by normal analysts in T hours is $T\mu$; that is, $\mu T - (T-\epsilon)\lambda$ of the additional alerts could be served by normal inspections. Thus, by the T^{th} hour, even if the defender has used all additional resources X, there are still $Y - X - \mu T - (T-\epsilon)\lambda$ alerts in the queue, which corresponds to $f(R - \mu T + (T-\epsilon)\lambda)$. Thus, the expected utility is worse than $-[1 - \exp(-\frac{\epsilon^2\lambda}{2T})] * f(R - \mu T + (T-\epsilon)\lambda)$. As an example, plugging in numbers $T = 14, \epsilon = 0.3, R = 4{,}800$, the queue length builds up to 4,210 with probability ≥ 0.998, thereby providing very low expected utility -0.95.

Next, we consider the case when $Y \leq X$. In this case, we first prove that the probability of the backlog being more than B over the finite horizon N is small.

A busy cycle for an M/D/1 queue is defined as a time period in which the queue length first goes to 0 starting from 0. It is known that all busy cycles are i.i.d., since an M/D/1 queue is a regenerative process [11]. A busy cycle with time length less than B/μ hour will not have more than B alerts in the queue, since, by definition, the server is busy and it can serve $< B$ alert in less than B/μ hour, so the max queue length cannot be $\geq B$. Thus, we focus on busy cycles of time length $\geq B/\mu$ hour. There can only be $N\mu/B$ such busy cycles in N hours.

In each busy cycle, the probability of the queue length being at most j is given by P_j, where it is known that $P_j \geq 1 - \frac{1}{j}$ [11]. Due to i.i.d. busy cycles, the probability of queue length being less than B over all the N hours is then more than $(P_B)^{N\mu/B}$.

Next, consider the event where the queue length remains below B for all N hours. Consider one run of the underlying queuing process with the queue lengths x_1 to x_{336} at every hour. Any additional alerts k sent by the adversary in the j^{th} hour raises the subsequent queue lengths to $x_i + k$ for $i \geq j$. Let the adversary send alerts quantity $k_1, k_2, \ldots$ at $j_1, j_2, \ldots$ respectively. According to the defender's policy, the queue length will always remain below B after the defender action as long as the defender has enough resources to allocate. Thus, we will argue that the defender resources are not exhausted. Consider the maximum queue length without defender intervention between j_1 and j_2: $Q = \max\{x_i + k_1 \mid j_1 \leq i < j_2\}$. Let the total defender intervention within j_1, j_2 be d_1. At any time, if the total defender intervention within j_1, j_2 is equal to $Q - B$, then the queue length remains below $\max\{x_i \mid j_1 \leq i < j_2\} \leq B$ beyond this point. The defender intervention also is never more than $Q - B$ because, if so, then the last time that the defender intervenes, he can reduce his resource allocation to be exactly $Q - B$ and achieve the goal of keeping the queue below B. Thus, $d_1 \leq Q - B$. Next, since each $x_i \leq B$, we have $k_1 \geq Q - B \geq d_1$. In a similar manner, it can be seen that $k_i \geq d_i$ for all i. Thus, the total additional resources used by the defender will always be less than the additional alerts sent by the adversary. Since the budgets are the same, the defender will not run out of resources.

Given the maximum queue length remains below B, providing defender utility $-f(B)$ with probability more than $(P_B)^{N\mu/B}$ and assuming the worst-case utility of -1 otherwise, we obtain the expected reward: $(1 + f(B))(P_B)^{N\mu/B} - 1$. For example, $B = 1{,}500$ yields around -0.3 utility. □

The analysis above reveals that there exists robust defender polices that are robust to any attacker policy only when the resources (budget) of the attacker are lower. Thus, robustness is highly dependent on the resources of the players; that is, the player with a resource advantage wins the game. The observed robustness of the CSOC-RL defender policy in experiments can be explained as the learned defender policy being one of the robust policies (or being close enough). As far as we know, this is the first analysis of a game theoretic model of a scenario when two opposing players want to control the queue length in an M/D/1 queue.

Further, the theorem provides a simple rule-based baseline policy $S1$. We further propose a more aggressive rule-based policy: ($S2$) whenever the backlog exceeds B by x, allocate $x + (B - L)$

additional inspections. $S2$ is more aggressive as it brings the backlog down to the acceptable lower bound L whenever it exceeds B as opposed to just bringing the backlog down to B. We compare the RL approach against both $S1$ and $S2$ in our experiments.

However, the theoretical result above is limited in a few ways. First, the result is proved under a fixed service rate. Second, the policy $S1$, while guaranteeing a minimum reward, is not necessarily the best policy. An RL policy that learns from experience and adapts more frequently than $S1$ might be better; in fact, Figures 2(f) and 3(c) show that our defender RL policy does better than $S1$. Third, the above analysis is under the modeling assumption that both defender and adversary actions are multiples of M. In the next section, we show how violating this assumption enables the adversary to overwhelm the queue. Finally, the theoretical analysis assumes simple mathematical properties such as Poisson arrivals; as such, $S1$ will do good only when such assumptions hold (which do hold for our experiments). On the other hand, an RL system will learn the arrival distribution and, accordingly, be robust to the variations in the arrival distribution.

4.3 Attacking Modeling Assumptions

The last two sections followed the assumption from the CSOC-RL work that the attacker generates alerts with the same discretization in the action-space as the defender, which was fixed to multiples of $M = 60$ alerts. This models a practical constraint for the defender that an additional analyst must be allocated for a minimum amount of time once engaged. However, this constraint in the action-space is too restrictive for the attacker. Thus, we relax this constraint for the attacker allowing him to have a finer discretization by sending alerts in multiples of 30. We attack using the same adversarial RL setup of Section 4.1 with this new chunk size of 30 for the adversary. We show in the experiments that this attack succeeds in building up a large backlog (high AvgTTA). The result is explained by the observation that the attacker uses few additional alerts to elicit a disproportionate response of additional resources from the defender. Such a disproportionate response is harmful for the defender, which can be seen in the following two scenarios that can arise: (1) the additional inspection resources (along with the baseline inspection resources arising from the service rate μ) reduce the queue length to zero and then some inspection resources (additional or baseline) are wasted; or (2) the additional resources are spent in handling the baseline alerts due to a sufficient number of baseline alerts already present in the queue, but later, when fewer baseline alerts arrive, some of the baseline inspection resources are wasted (because a portion of the queue is already cleared by the additional inspections). In either case, as more additional inspections are spent than the additional alerts generated, at some point, all additional inspections will be exhausted with many additional alerts still available for the adversary to generate. At this point, the adversary can use the leftover additional alerts to increase the queue length since the baseline inspection rate μ is only enough to handle the baseline alert generation at rate λ. We also experimented with smaller chunks than 30 for the attacker.

The fix for the above attack is inferred from the game theoretic nature of the problem. In particular, this new attack policy is an adversary best response that was not considered when learning the defender RL policy. Thus, using a *double oracle* like approach, we retrain the defender by including episodes from this new attacker policy in the defender RL training. The procedure is precisely described in Algorithm 1. After the first defender attacker policy is trained, it is checked whether the new attacker policy is successful (line 5). If the attack is successful, then the simulator for the defender RL environment is updated to use the new attacker policy (line 9; see also Figure 1). Then, in the next run of the loop, the defender RL training is done again with the RL environment now using the new attacker policy. Similarly, the RL environment for the attacker is updated with the new, retrained, defender policy (line 4, see also Figure 1) and the attacker policy is relearned (line 5). This keeps continuing in the while loop till no new attacks are discovered. For the finer

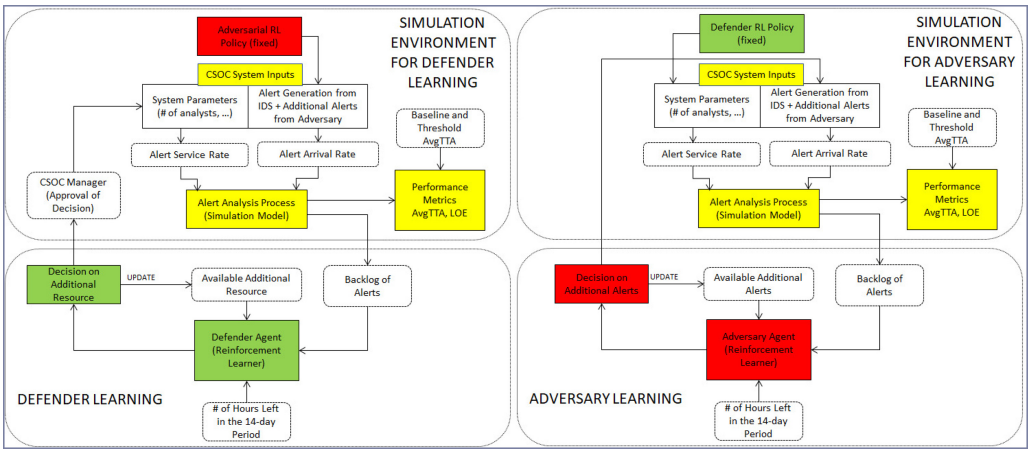


Fig. 1. A visual representation of the defender and attacker learning.

ALGORITHM 1: CSOC-RL Algorithm

Input: All parameters for the generation of non-adversarial alerts as in earlier work [36]
Output: Defender policy τ.

```
Build initial simulator for defender RL as described in Reference [36];
while new_attack = true do
    Perform defender RL training as described in Algorithm 2 of Reference [36] to get the defender
    policy τ;
    Use the defender policy τ as the fixed defender policy to build a simulator (the MDP environment)
    for the adversary RL as described in Section 4.1.;
    Perform adversary RL training as described in Section 4.1 to get adversary policy σ;
    Simulate the defender policy τ and adversary policy σ and obtain the results for queue lengths (as
    in Figure 4(a) and (c));
    if the queue length results show an attack then
        new_attack = true;
        Update the simulator for defender RL by substituting the adversary's action by the new
        adversary policy σ;
    else
        new_attack = false
    end
end
return τ
```

discretization attack discovered, we find that the new relearned defender policy is robust to the discovered attacker policy after just one relearning of the defender policy; then, no new harmful attacker policies were discovered when the adversarial RL approach was used to attack the relearned defender policy.

We call the above method "double oracle like" as the classic double oracle approach [26] involves computing the game equilibrium in each iteration round (while loop) with all discovered players' policy till that iteration, which gives a mix (probability distribution) of multiple policies as the resultant strategy of a player at that step. In contrast, we aim to find one defender policy that is robust to all successful attacker policies. Our experiments reveal that we are indeed able to find

such a defender policy for our problem. Theorem 1 provides a mathematical explanation (for a restricted case though) for the existence of such defender policies in our problem setting.

5 EXPERIMENTS

We explain the experimental setup including our measurement metrics. First, the experimental setup is to simulate the arrival and processing of alerts. The simulation model we use is the same as in the previous CSOC-RL work [37]. At a high level, the simulator simulates the underlying queuing process as well as uncertain events such as analysts absenteeism and the like. Other significant realistic aspects simulated are that additional analyst's time is obtained by first making regular analysts work overtime and then bringing in additional analysts, if required. As in the previous work, we fix the defender budget as 28,800 additional inspections over the $N = 336$ hours.

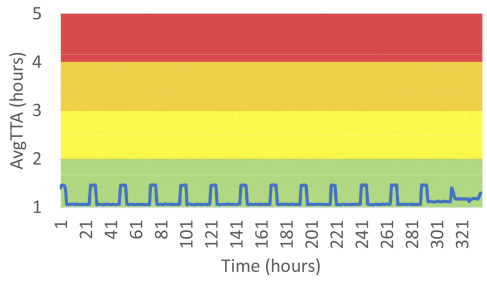
In the same CSOC-RL work, the authors rely on a metric called average total time for alert investigation (AvgTTA). The time for alert investigation of an alert is the waiting time in the queue and the analyst investigation time after its arrival in the CSOC database. The AvgTTA is estimated as the average of the time for alert investigation values of all the alerts that were investigated in an hour. It is to be noted that the AvgTTA is measured on an hourly basis in the experiments. It is a requirement of the CSOC under study that the AvgTTA remain within 4 hours. One hour or less was determined to be ideal, and anything within 1–4 hours is acceptable.

The CSOC-RL application uses a color-coded representation of AvgTTA, which was developed in a prior work [37]. Different color-coded tolerance bands are created below the 4 hours and above the 1 hour value of AvgTTA, for example, as shown in Figure 2(a). As stated earlier, the AvgTTA value directly corresponds to the amount of backlog (1 hour is 1,175 alerts, 4 hours is 4,350 alerts, and is linearly interpolated). The color-coded representation of AvgTTA is as follows: 1–2 hours is green (acceptable), followed by yellow for 2–3 hours (acceptable), orange for 3–4 hours (acceptable), and red (unacceptable) above 4 hours. Any value below 1 hour is rounded up to 1 hour. Our results show the hour by hour backlog for the worst run (run with maximum backlog ever) among 500 runs using the color-coded representation (for example, the line in Figure 2(a)). It is to be noted that the worst run corresponds to the run with the maximum amount of backlog observed between time t and $t + 1$. We decided to use the worst run in place of the average values for each timestamp so that we can obtain a temporal observation of the CSOC performance in that respective run. It is known that interpretability and usable interfaces are among the main issues in the adoption of AI technologies in the real world [33, 39]. Toward that end, the color-coded visualization was found to be extremely useful in displaying and explaining the CSOC-RL results to non-mathematical experts [36]. While the visualization is coarser than numerical results expressed using the function f, they are much more easily interpretable by humans.

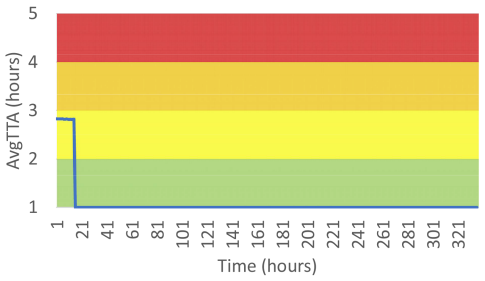
In this work, we introduce another easily interpretable metric that we call the AvgTTA proportions. Given the stochastic arrivals of alerts due to the queuing process, it is not sufficient to look at a worst-case run of the system. Thus, we show a pie-chart with the proportion of hours among the $500N$ hours over 500 distinct runs corresponding to backlog in one of the four color bands.

5.1 Analyst Scheduling and Experimental Setup

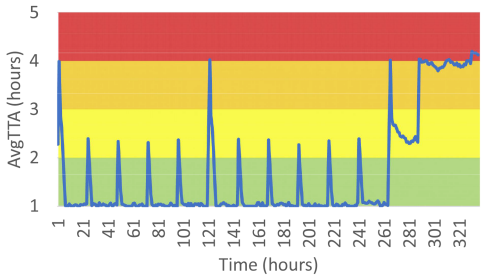
Next, we present an overview of the analyst scheduling and experimental setup. We follow the same setup as in the prior work [36], except for the intelligent adversary part, which we specify in subsequent sections. Table 1 shows the parameters used for the experiments. The base alerts are generated stochastically as a Poisson process from the sensor clusters with the average time between alert generation given in Table 1. Note that additional alerts are generated by the adversary. The alert analysis process at the CSOC is simulated using the algorithm in Reference [37]. It is to be noted that the analysts are staffed such that the service rate of alerts is slightly higher than the



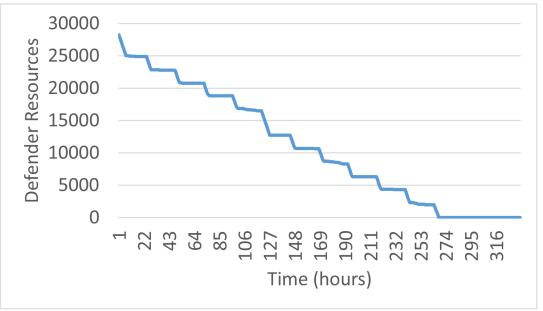
(a) Daily bound worst run



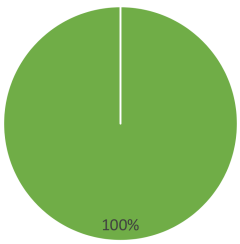
(b) Unbounded worst run



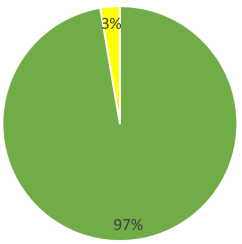
(c) 10% extra budget worst run



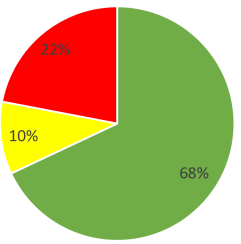
(d) Defender additional resources for Fig. 2(c)



(e) Daily bound proportions



(f) Unbounded proportions



(g) 10% extra budget proportions

Fig. 2. Results for Adversarial RL attacks.

Table 1. Parameter Values for the CSOC-RL Model

Parameter	Value
Number of clusters of sensors	10
Distribution of time between alert generation (s)	Exponential(18.8)
Base number of analysts	10
% effort of analysts toward alert analysis	80%
Average time taken to investigate an alert (s)	15
Baseline AvgTTA (in hours)	1
Threshold AvgTTA (in hours)	4

arrival rate of alerts, i.e., $\rho = (\lambda/\mu) < 1$. It is customary at a CSOC that an analyst's time is split between alert analysis work, report writing, training, and on generating signatures for Intrusion Detection System (IDS). Based on the work in Shah et al. [36], we keep the percentage effort of all the analysts toward alert analysis work at 80%. It is to be noted that the alert service time (μ) is calculated using the time allocated to the alert analysis process (i.e., 80%). Upon making the decision to allocate resources, the CSOC first utilizes the remaining 20% of the analysts' time available during the work-shift to analyze alerts in the backlog queue before summoning the on-call analysts.

5.2 Adversarial RL

We attack the learned defender RL using our adversarial RL method stated earlier. We try three different variations on the adversary budget: (1) the adversary budget is the same as the defender budget, but is constrained by a daily bound $28{,}800/14 \approx 2{,}057$ of additional alerts, (2) the adversary budget same is the same as the defender budget with no bounds, and (3) the adversary budget is 10% more than the defender budget with daily bounds.

The results for the worst-case (maximum overall backlog) run of the system as well the AvgTTA distribution are shown in Figure 2. The adversary with the same budget and a daily bound (case 1 above) is unable to push the backlog beyond the green band in any of the 500 runs (Figure 2(e)) with the daily bounds. Figure 2(a) shows that the attacker dumps all additional alerts allowed in a day at the start of the day. Even with no bounds, the attacker with the same budget (case 2 above) is unable to push the backlog beyond the yellow band in the worst case (Figure 2(b)), with 97% of $500N$ hours remaining in the green band (Figure 2(f)). Again, all additional alerts are dumped in a few hours at the start of the time horizon.

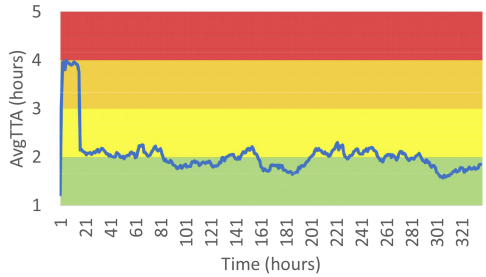
However, with just an extra 10% budget, even the daily bounded attacker (case 3 above) with a daily bound $1.1 * 28{,}800/14 \approx 2{,}262$ is able to push the backlog into the red zone in 22% of the $500N$ hours (Figure 2(g)); Figure 2(c) shows that toward the end, the backlog builds up and stays high since the defender resources get exhausted. Figure 2(d) shows the defender resources getting exhausted before N hours corresponding to the runs in Figure 2(c). The backlog does not reach the red zone always, because in those runs where the number of stochastic alerts is low, the defender does not expend its additional resources, and is able to control the backlog.

These results provide an empirical evidence that the learned defender RL policy is robust to the best response of the attacker. Further, the results also reveal the subtle relation between the robustness of the defender and the budgets of players as claimed in Theorem 1.

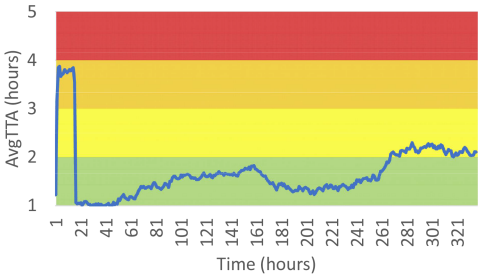
5.3 Theoretical Baseline

Our next set of experiments are for the baselines $S1$ and $S2$ that we derived in Section 4.2. Recall from Theorem 1 that B is the threshold parameter for policy $S1$. We choose the threshold B corresponding to AvgTTA of 2 hours, or equivalently 2,233 alerts. We experiment with the more powerful unbounded attacker (i.e., no daily bound) with the same budget as the defender budget. The worst-case runs (Figure 3(a) and (b)) show that the baselines perform reasonably and are able to recover from the initial barrage of alerts sent by the unbounded attacker. Yet, the backlog still goes to the orange band as can be seen from the figures. Further, compared to the proportions for RL policy (Figure 2(f)), the proportions are worse for the rule-based policy (Figure 3(c) and (d)). As pointed out in the discussion after Theorem 1, this shows that the theoretically rule-based policy is not necessarily the best robust policy.

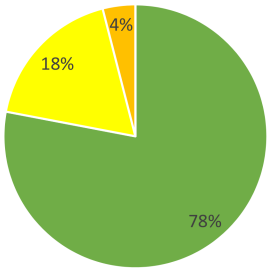
However, the rule-based policy is much simpler and more interpretable than a large RL policy. In literature, there is evidence for the phenomenon that sub-optimal but simple decision aids are more convincing for human users and more likely to be trusted and adopted [15]. Thus, while sub-optimal, the rule-based policy is still a competitive candidate for deployment.



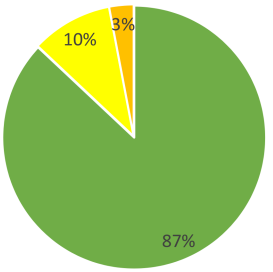
(a) $S1$ policy worst run



(b) $S2$ policy worst run



(c) $S1$ policy proportions



(d) $S2$ policy proportions

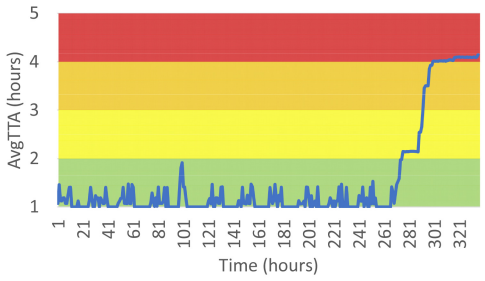
Fig. 3. Results for attack on baseline policies $S1$ and $S2$.

5.4 Attacking Modeling Assumptions

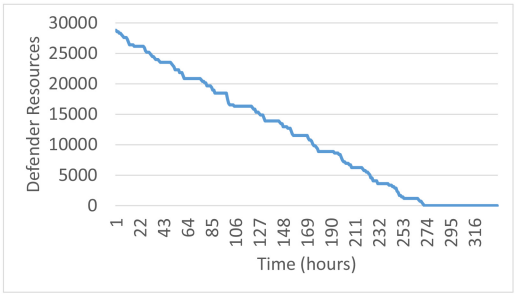
Finally, we show results for our attack that exploited modeling assumptions in the prior CSOC-RL work. We conduct this attack with the restricted attacker (with daily bounds) and the same budget as the defender budget. We allow the attacker to choose a chunk size of 30. Figure 4(a) shows that even the restricted attacker with a daily bound is able to successfully push the backlog into the red zone. We observe that the attacker is able to elicit a disproportionate response by the defender, which results in the defender's additional resources getting exhausted toward the end of the time horizon (please see the intuitive explanation for this provided in Section 4.3); Figure 4(b) shows the defender resources getting exhausted before N hours corresponding to the runs in Figure 4(a). Overall, the attacker is able to keep the backlog in the red zone in 12% of the $500N$ hours (Figure 4(c)).

The performance of the corresponding robust defender policy learned by retraining the defender with episodes from the adversarial attack is shown in Figure 4(d) and (e). Intuitively, the relearned defender policy exhibits more patience by allowing the backlog to be higher on average (worst run in Figure 4(e)), yet manages to keep the backlog in the green zone over the time horizon for 95% of the $500N$ hours (piechart in Figure 4(d)). We also attacked the relearned defender policy using various different chunk sizes for the adversary (all the way down to one) and found the defender to be robust against all of the chunk sizes less than 30.

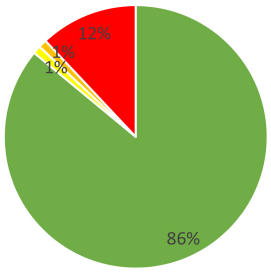
Next, we conduct the same model assumption attacks against the baselines $S1$ and $S2$. Figure 5 shows the results of attacking $S1$ and $S2$ using the restricted daily bounded adversary with a finer discretization of size 30. It can be seen that $S1$ is quite robust to the attack, whereas the more aggressive $S2$ suffers from the attack, which highlights our observation that a more patient policy



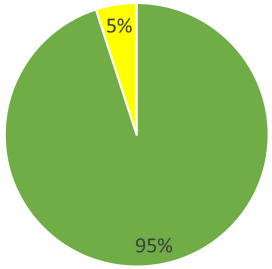
(a) Model assumption attack worst run



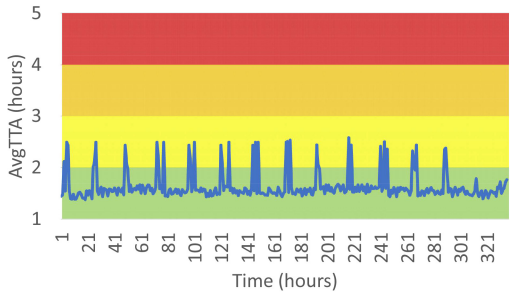
(b) Defender resources for Fig. 4(a)



(c) Model assumption attack proportions



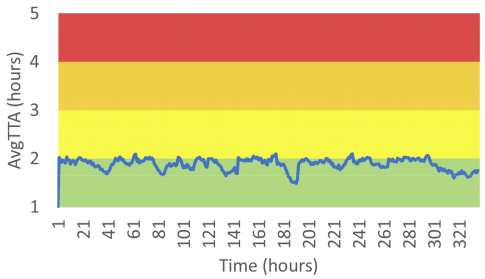
(d) Retrained robust defender proportions



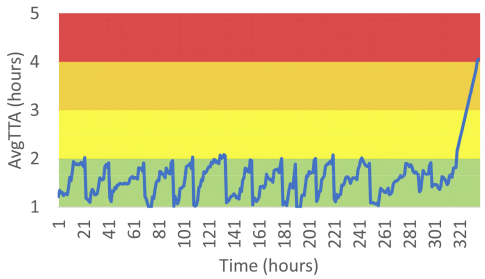
(e) Retrained robust defender worst run

Fig. 4. Results for attack on baseline policies $S1$, $S2$ and attack against model assumption.

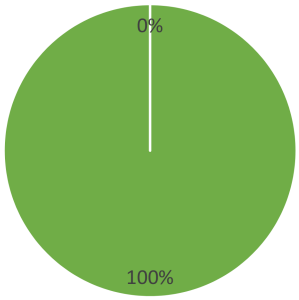
is beneficial against this attack. For $S2$, the defender resources get exhausted before the N hours, and then the attacker is able to successfully push the backlog high. Note that the piechart for $S1$ looks better than Figure 3(c) because this attack used a daily bounded adversary, unlike the unbounded adversary attack on $S1$ and $S2$ in Figure 3(c) and (d) respectively. Also, observe that the relearned defender policy, while going to the yellow zone about 5% of the time, keeps the average queue length lower than the average queue length for $S1$; thus, on average, it is better than $S1$. As stated after Theorem 1, the theoretical analysis was done using a modeling assumption that both defender action (additional resources) and adversary action (additional alerts) are multiples of M, which followed the model from the prior CSOC-RL work. The experimental results we showed here reveal that in spite of the theoretical defense guarantee provided in Theorem 1, an attacker can succeed in overwhelming the queue by exploiting such assumptions built into the mathematical model.



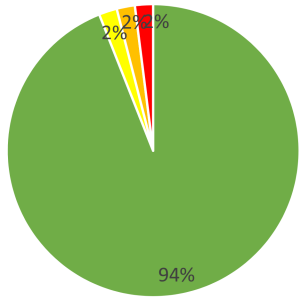
(a) Modeling assumption attack ($S1$) worst run



(b) Modeling assumption attack ($S2$) worst run



(c) Modeling assumption attack ($S1$) proportions



(d) Modeling assumption attack ($S2$) proportions

Fig. 5. Modeling assumption attack on $S1$ and $S2$.

6 CONCLUSION

We performed a red team evaluation of a cyber-alert inspection system that is under consideration for deployment. We observed the system to be robust to the best responding adversarial alert generation policy, but we also showed that a weakness in the model assumption could be exploited by an attacker. We provided a game theoretic formulation of the problem, which allowed us to understand the defender-adversary interaction. In particular, we showed that players' resources (budget) are a critical factor in deciding whether the defender policy can be robust. The theory also yielded simple and sub-optimal but usable defender policies. Further, using game theoretic insight we made the defender RL approach robust to the discovered successful attacker policy. The adversarial RL and double oracle approach in RL are general techniques that are applicable to other RL usage in adversarial environments.

REFERENCES

[1] Bob Ackerman. 2017. The healthcare industry is in a world of cybersecurity hurt. Retrieved August 10, 2018 from https://tcrn.ch/2OqmOX9.techcrunch.com.
[2] Eitan Altman. 2005. *Applications of Dynamic Games in Queues*. Birkhäuser Boston, 309–342.
[3] Itai Ashlagi, Brendan Lucier, and Moshe Tennenholtz. 2013. Equilibria of online scheduling algorithms. In *AAAI*.
[4] Vahid Behzadan and Arslan Munir. 2017. Vulnerability of deep reinforcement learning to policy induction attacks. In *International Conference on Machine Learning and Data Mining in Pattern Recognition*. Springer, 262–275.
[5] Richard Bejtlich. 2005. *The Tao of Network Security Monitoring: Beyond Intrusion Detection*. Pearson Education Inc.
[6] Sandeep Bhatt, Pratyusa K. Manadhata, and Loai Zomlot. 2014. The operational role of security information and event management systems. *IEEE Security & Privacy* 12, 5 (2014), 35–41.
[7] Jeremiah Blocki, Nicolas Christin, Anupam Datta, Ariel D. Procaccia, and Arunesh Sinha. 2013. Audit games. In *Proceedings of IJCAI*.
[8] Matthew Brown, Arunesh Sinha, Aaron Schlenker, and Milind Tambe. 2016. One size does not fit all: A game-theoretic approach for dynamically and effectively screening for threats. In *Proceedings of AAAI*.

[9] Corbin Carlo. 2003. Intrusion detection evasion: How Attackers get past the burglar alarm. Retrieved February 10, 2019 from https://www.sans.org/reading-room/whitepapers/detection/intrusion-detection-evasion-attackers-burglar-alarm-1284. SANS Institute.
[10] Jacob W. Cohen. 2012. *On Regenerative Processes in Queueing Theory*. Vol. 121. Springer Science & Business Media.
[11] Jacob Willem Cohen and Anthony Browne. 1982. *The Single Server Queue*. Vol. 8. North-Holland Amsterdam.
[12] Tim Crothers. 2002. *Implementing Intrusion Detection Systems*. Wiley Publishing Inc.
[13] Anita D'Amico and Kirsten Whitley. 2008. *VizSEC 2007: Proceedings of the Workshop on Visualization for Computer Security*. Springer Berlin, 19–37. DOI: https://doi.org/10.1007/978-3-540-78243-8_2
[14] Karel Durkota, Viliam Lisy, Branislav Bošansky, and Christopher Kiekintveld. 2015. Optimal network security hardening using attack graph games. In *International Conference on Artificial Intelligence*. 526–532.
[15] Avshalom Elmalech, David Sarne, Avi Rosenfeld, and Eden Shalom Erez. 2015. When suboptimal rules. In *Proceedings of AAAI*.
[16] Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. 2017. Optimal scheduling of cybersecurity analyst for minimizing risk. *ACM Transactions on Intelligent Systems and Technology* 8, 4, Article 52 (Feb. 2017), 1–32. http://dx.doi.org/10.1145/2914795
[17] Hugo Gimbert, Jérôme Renault, Sylvain Sorin, Xavier Venel, and Wiesław Zielonka. 2016. On values of repeated games with signals. *The Annals of Applied Probability* 26, 1 (2016), 402–424.
[18] Marek Grześ. 2017. Reward shaping in episodic reinforcement learning. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, 565–573.
[19] Eric A. Hansen, Daniel S. Bernstein, and Shlomo Zilberstein. 2004. Dynamic programming for partially observable stochastic games. In *Proceedings of the 19th National Conference on Artificial Intelligence*. 709–715.
[20] Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. 2017. Adversarial attacks on neural network policies. *arXiv preprint arXiv:1702.02284* (2017).
[21] Marc Lanctot, Vinicius Zambaldi, Audrunas Gruslys, Angeliki Lazaridou, Karl Tuyls, Julien Pérolat, David Silver, and Thore Graepel. 2017. A unified game-theoretic approach to multiagent reinforcement learning. In *Advances in Neural Information Processing Systems*. 4190–4203.
[22] Aron Laszka, Jian Lou, and Yevgeniy Vorobeychik. 2016. Multi-defender strategic filtering against spear-phishing attacks. In *AAAI*.
[23] Shuxin Li, Xiaohong Li, Jianye Hao, Bo An, Zhiyong Feng, Kangjie Chen, and Chengwei Zhang. 2017. Defending against man-in-the-middle attack in repeated games. In *International Joint Conference on Artificial Intelligence*.
[24] Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. 2017. Tactics of adversarial attack on deep reinforcement learning agents. In *Proceedings of ICJAI*.
[25] Carl Manion. 2016. How to Avoid Wasting Time on False Positives. Retrieved August 10, 2018 from https://www.rsaconference.com/blogs/how-to-avoid-wasting-time-on-false-positives. Raytheon Foreground Security.
[26] H. Brendan McMahan, Geoffrey J. Gordon, and Avrim Blum. 2003. Planning in the presence of cost functions controlled by an adversary. In *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*. 536–543.
[27] Jean-FranÃğois Mertens, Sylvain Sorin, and Shmuel Zamir. 2015. *Repeated Games*. Cambridge University Press.
[28] Andrew Y. Ng, Daishi Harada, and Stuart Russell. 1999. Policy invariance under reward transformations: Theory and application to reward shaping. In *ICML*, Vol. 99. 278–287.
[29] Mina Guirguis Noah Dunstatter and Alireza Tahsini. 2018. Allocating security analysts to cyber alerts using Markov games. In *Proceedings of National Cyber Summit*.
[30] Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. 2017. Robust adversarial reinforcement learning. In *International Conference on Machine Learning*. 2817–2826.
[31] David Pollard. 2015. A few good inequalities. Retrieved from http://www.stat.yale.edu/ pollard/Books/Mini/Basic.pdf.
[32] Amin Rasoulifard, Abbas Ghaemi Bafghi, and Mohsen Kahani. 2008. Incremental hybrid intrusion detection using ensemble of weak classifiers. In *Advances in Computer Science and Engineering*. Springer, 577–584.
[33] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. Why should I trust you?: Explaining the predictions of any classifier. In *SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM.
[34] Aaron Schlenker, Omkar Thakoor, Haifeng Xu, Fei Fang, Milind Tambe, Long Tran-Thanh, Phebe Vayanos, and Yevgeniy Vorobeychik. 2018. Deceiving cyber adversaries: A game theoretic approach. In *International Conference on Autonomous Agents and MultiAgent Systems*.
[35] Aaron Schlenker, Haifeng Xu, Mina Guirguis, Chris Kiekintveld, Arunesh Sinha, Milind Tambe, Solomon Sonya, Darryl Balderas, and Noah Dunstatter. 2017. Don't bury your head in warnings: A game-theoretic approach for intelligent allocation of cyber-security alerts. In *Proceedings of IJCAI*.
[36] Ankit Shah, Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. 2018. Dynamic optimization of the level of operational effectiveness of a CSOC under adverse conditions. *ACM Transactions on Intelligent Systems and Technology* 9, 5 (2018), 1–20.

[37] Ankit Shah, Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. 2018. A methodology to measure and monitor level of operational effectiveness of a CSOC. *International Journal of Information Security* 17, 2 (Apr. 2018), 121–134.
[38] Ankit Shah, Rajesh Ganesan, Sushil Jajodia, and Hasan Cam. 2019. Understanding tradeoffs between throughput, quality, and cost of alert analysis in a CSOC. *IEEE Transactions on Information Forensics and Security* 14, 5 (2019), 1155–1170.
[39] Sebastian Stein, Enrico H. Gerding, Adrian Nedea, Avi Rosenfeld, and Nicholas R. Jennings. 2017. Market interfaces for electric vehicle charging. *Journal of Artificial Intelligence Research* 59 (2017), 175–227.
[40] Kimberly Underwood. 2017. Cyber Attacks on Government Agencies on the Rise. Retrieved August 10, 2018 from https://www.afcea.org/content/cyber-attacks-government-agencies-rise.
[41] Yufei Wang, Zheyuan Ryan Shi, Lantao Yu, Yi Wu, Rohit Singh, Lucas Joppa, and Fei Fang. 2018. Deep reinforcement learning for green security games with real-time information. 33, 1 (2018), 1401–1408.
[42] Chao Yan, Bo Li, Yevgeniy Vorobeychik, Aron Laszka, Daniel Fabbri, and Bradley Malin. 2018. Get your workload in order: Game theoretic prioritization of database auditing. In *International Conference on Data Engineering*.
[43] Mengchen Zhao, Bo An, and Christopher Kiekintveld. 2016. Optimizing personalized email filtering thresholds to mitigate sequential spear phishing attacks. In *AAAI*.

Received April 2019; revised November 2019; accepted December 2019