Learning Security Strategies through Game Play and Optimal Stopping

Kim Hammar [†‡] and Rolf Stadler[†‡]
[†] Division of Network and Systems Engineering, KTH Royal Institute of Technology, Sweden
[‡] KTH Center for Cyber Defense and Information Security, Sweden
Email: {kimham, stadler}@kth.se

May 27, 2022

***Abstract*—We study automated intrusion prevention using reinforcement learning. Following a novel approach, we formulate the interaction between an attacker and a defender as an optimal stopping game and let attack and defense strategies evolve through reinforcement learning and self-play. The game-theoretic perspective allows us to find defender strategies that are effective against dynamic attackers. The optimal stopping formulation gives us insight into the structure of optimal strategies, which we show to have threshold properties. To obtain the optimal defender strategies, we introduce T-FP, a fictitious self-play algorithm that learns Nash equilibria through stochastic approximation. We show that T-FP outperforms a state-of-the-art algorithm for our use case. Our overall method for learning and evaluating strategies includes two systems: a simulation system where defender strategies are incrementally learned and an emulation system where statistics are produced that drive simulation runs and where learned strategies are evaluated. We conclude that this approach can produce effective defender strategies for a practical IT infrastructure.**

***Index Terms*—Network security, automation, optimal stopping, reinforcement learning, game theory, Markov decision process, Dynkin games, MDP, POMDP**

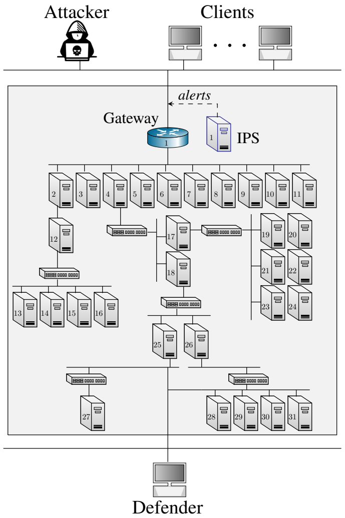

Fig. 1: The IT infrastructure and the actors in the use case.

I. Introduction

An organization's security strategy has traditionally been defined, implemented, and updated by domain experts [1]. Although this approach can provide basic security for an organization's communication and computing infrastructure, a growing concern is that infrastructure update cycles become shorter and attacks increase in sophistication [2]. Consequently, the security requirements become increasingly difficult to meet. To address this challenge, significant efforts have started to automate security frameworks and the process of obtaining security strategies. Examples of this research include: automated creation of threat models [3]; computation of defender strategies using dynamic programming and control theory [4], [5]; computation of exploits and corresponding defenses through evolutionary methods [6]; identification of infrastructure vulnerabilities through attack simulations and threat intelligence [7], [8]; computation of defender strategies through game-theoretic methods [9], [10]; and use of machine learning techniques to estimate model parameters and strategies [11], [12], [13].

In this paper, we present a novel approach to automatically learn security strategies for an attacker and a defender. We apply this approach to an *intrusion prevention* use case, which involves the IT infrastructure of an organization (see Fig. 1). The operator of this infrastructure, which we call the defender, takes measures to protect it against a possible attacker while providing services to a client population. (We use the term ”intrusion prevention” as suggested in the literature, e.g. in [1], it means that the attacker is prevented from reaching its goal, rather than prevented from accessing any part of the infrastructure.)

We formulate the use case as an *optimal stopping game*, i.e. a stochastic game where each player faces an optimal stopping problem [14], [15], [16]. The stopping game formulation enables us to gain insight into the structure of optimal strategies, which we show to have threshold properties. To obtain effective defender strategies, we use reinforcement learning and self-play. Based on the threshold properties of optimal strategies, we design an efficient self-play algorithm that iteratively computes optimal defender strategies against a dynamic attacker.

Our method for learning and evaluating strategies includes building two systems (see Fig. 2). First, we develop an *emulation system* where key functional components of the target infrastructure are replicated. This system closely approximates the functionality of the target infrastructure and is used to run attack scenarios and defender responses. These

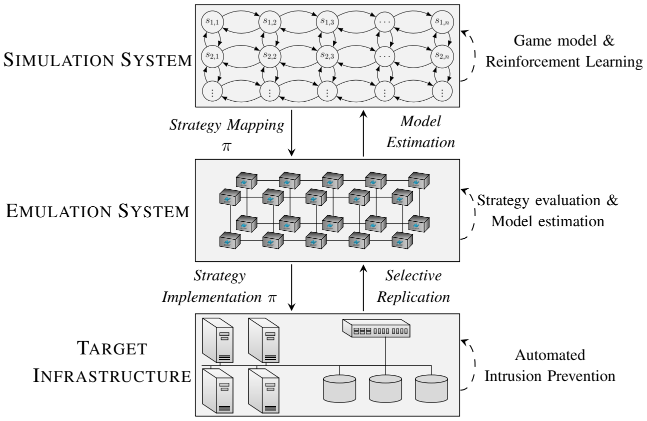

Fig. 2: Our approach for finding and evaluating intrusion prevention strategies.

runs produce system measurements and logs from which we estimate distributions of infrastructure metrics. We then use the estimated distributions to instantiate the simulation model. Second, we build a *simulation system* where game episodes are simulated and strategies are incrementally learned. Learned strategies are then extracted from the simulation system and evaluated in the emulation system. (A video demonstration of our software framework that implements the emulation and simulation systems is available at [17].)

We make three contributions with this paper. First, we formulate intrusion prevention as an optimal stopping game. This novel formulation allows us a) to derive structural properties of optimal strategies using results from optimal stopping theory; and b) to find defender strategies that are effective against attackers with dynamic strategies. We thus address a limitation of many related works that consider static attackers only [18], [19], [20], [21], [22], [13], [23], [24]. Second, we propose T-FP, an efficient reinforcement learning algorithm that exploits structural properties of optimal stopping strategies and outperforms a state-of-the-art algorithm for our use case. Third, we provide evaluation results from an emulated infrastructure. This addresses a common drawback in related research, which relies on simulations to learn and evaluate strategies [12], [13], [25], [26], [18], [24], [27], [19], [28], [20], [21], [23], [10], [29], [30], [31].

II. The Intrusion Prevention Use Case

We consider an intrusion prevention use case that involves the IT infrastructure of an organization. The operator of this infrastructure, which we call the defender, takes measures to protect it against an attacker while, at the same time, providing a service to a client population (Fig. 1). The infrastructure includes a set of servers that run the service and an Intrusion Prevention System (IPS) that logs events in real-time. Clients access the service through a public gateway, which also is open to the attacker.

The attacker's goal is to intrude on the infrastructure and compromise a set of its servers. To achieve this, the attacker must explore the infrastructure through reconnaissance and exploit vulnerabilities, while, at the same time, avoid getting detected by the defender. The attacker decides when to start the intrusion and may also decide to stop the intrusion at any moment. During the intrusion, it is assumed that the attacker follows a pre-defined strategy. When deciding the time to start and stop the intrusion, the attacker needs to consider both the gain of compromising additional servers in the infrastructure and the risk of getting detected by the defender. The optimal strategy for the attacker is to compromise as many servers as possible without getting detected.

The defender continuously monitors the infrastructure through accessing and analyzing IPS statistics. It can take a fixed number of defensive actions, each of which has a cost and a chance of preventing the attacker. A defensive action is for example to configure the IPS to drop network traffic that triggers alerts of a certain priority. It is assumed that the defender takes the defensive actions in a pre-determined order, starting with the action that has the lowest cost. The final action that the defender can take is to block all external access to the gateway. As a consequence of this action, the service as well as any ongoing intrusion are disrupted.

In deciding when to take defensive actions, the defender has two objectives: (*i*) maintain service to its clients; and (*ii*), keep a possible attacker out of the infrastructure at low cost. The optimal strategy for the defender is to monitor the infrastructure and maintain service until the moment when the attacker enters through the gateway, at which time the attacker must be prevented with the minimal cost by taking defensive actions. The challenge for the defender is to identify the precise time when this moment occurs.

III. Formalizing The Intrusion Prevention Use Case

We model the use case as a partially observed stochastic game. The attacker wins the game when it can intrude on infrastructure and hide its actions from the defender. In contrast, the defender wins the game when it manages to prevent an intrusion. We model this as a zero-sum game, which means that the gain of one player equals the loss of the other player.

The attacker and the defender have different observability in the game. The defender observes alerts from an Intrusion Prevention System (IPS) but has no certainty about the presence of an attacker or the state of a possible intrusion. The attacker, on the other hand, is assumed to have complete observability. It has access to all the information that the defender has access to, as well as the defender's past actions. The asymmetric observability requires the defender to find strategies that are effective against any attacker, including attackers with inside information about its monitoring capabilities.

The reward function of the game encodes the defender's objective. An optimal defender strategy *maximizes* reward when facing a worst-case attacker. Similarly, an optimal attacker strategy *minimizes* reward when facing a worst-case defender. In game-theoretical terms, this means that a pair of optimal strategies is a Nash equilibrium [32].

A. The Intrusion Prevention Game

We model the intrusion prevention use case as a finite and zero-sum Partially Observed Stochastic Game (POSG) with one-sided partial observability: $\Gamma = \langle \mathcal{N}, \mathcal{S}, (\mathcal{A}_i)_{i\in\mathcal{N}}, \mathcal{T}, (\mathcal{R}_i)_{i\in\mathcal{N}}, \gamma, \rho_1, T, (\mathcal{O}_i)_{i\in\mathcal{N}}, \mathcal{Z}\rangle$. It is a discrete-time game that starts at time $t = 1$. In the following, we describe the components of the game, its evolution, and the objectives of the players.

Players $\mathcal{N}$: The game has two players: player 1 is the defender and player 2 is the attacker.

State space $\mathcal{S}$: The game has three states: $s_t = 0$ if no intrusion is occurring, $s_t = 1$ if an intrusion is ongoing, and $s_t = \emptyset$ if the game has ended. The initial state is $s_1 = 0$ and the initial state distribution is the degenerate distribution $\rho_1(0) = 1$.

Action spaces $\mathcal{A}_i$: Each player can invoke two actions: "stop" (S) and "continue" (C). The action spaces are thus $\mathcal{A}_1 = \mathcal{A}_2 = \{S, C\}$. S results in a change of state and C means that the game remains in the same state. We encode S with 1 and C with 0.

The attacker can invoke the stop action two times: the first to start the intrusion and the second to stop it. The defender can invoke the stop action $L \geq 1$ times. Each stop of the defender can be interpreted as a defensive action against a possible intrusion. The number of stops remaining of the defender at every time-step is known to both the attacker and the defender and is denoted by $l_t \in \{1, \dots, L\}$.

At each time-step, the attacker and the defender simultaneously choose an action each: $\boldsymbol{a}_t = (a_t^{(1)}, a_t^{(2)})$, where $a_t^{(i)} \in \mathcal{A}_i$.

Observation space $\mathcal{O}$: The attacker has complete information and knows the game state, the defender's actions, and the defender's observations. The defender, however, only sees the observations $o_t \in \mathcal{O}$, where $\mathcal{O}$ is a discrete set. (In our use case, o_t relates to the number of IPS alerts during time-step t.)

Both players have perfect recall, meaning that they remember their respective play history. The history of the defender at time-step t is $h_t^{(1)} = (\rho_1, a_1^{(1)}, o_1, \dots, a_{t-1}^{(1)}, o_t)$ and the history of the attacker is $h_t^{(2)} = (\rho_1, a_1^{(1)}, a_1^{(2)}, o_1, s_1, \dots, a_{t-1}^{(1)}, a_{t-1}^{(2)}, o_t, s_t)$.

Belief space $\mathcal{B}$: Based on its history $h_t^{(1)}$, the defender forms a belief about s_t, which is expressed in the *belief state* $b_t(s_t) = \mathbb{P}[s_t | h_t^{(1)}] \in \mathcal{B}$. Since $s_t \in \{0, 1\}$ and $b_t(0) = 1 - b_t(1)$, b_t is determined by $b_t(1)$. Hence, we can model $\mathcal{B} = [0, 1]$.

Transition probabilities $\mathcal{T}$: At each time-step t, a state transition occurs. The probabilities of the state transitions are defined by $\mathcal{T}_{l_t}\big(s_{t+1}, s_t, (a_t^{(1)}, a_t^{(2)})\big) = \mathbb{P}_{l_t}\big[s_{t+1} | s_t, (a_t^{(1)}, a_t^{(2)})\big]$:

$$\mathcal{T}_{l_t>1}\big(0, 0, (S, C)\big) = \mathcal{T}_{l_t}\big(0, 0, (C, C)\big) = 1 \tag{1}$$

$$\mathcal{T}_{l_t>1}\big(1, 1, (\cdot, C)\big) = \mathcal{T}_{l_t}\big(1, 1, (C, C)\big) = 1 - \phi_{l_t} \tag{2}$$

$$\mathcal{T}_{l_t>1}\big(1, 0, (\cdot, S)\big) = \mathcal{T}_{l_t}\big(1, 0, (C, S)\big) = 1 \tag{3}$$

$$\mathcal{T}_{l_t>1}\big(\emptyset, 1, (\cdot, C)\big) = \mathcal{T}_{l_t}\big(\emptyset, 1, (C, C)\big) = \phi_{l_t} \tag{4}$$

$$\mathcal{T}_1\big(\emptyset, \cdot, (S, \cdot)\big) = \mathcal{T}_{l_t}(\emptyset, \emptyset, \cdot) = \mathcal{T}_{l_t}(\emptyset, 1, (\cdot, S)) = 1 \tag{5}$$

All other state transitions have probability 0.

Eqs. 1-2 define the probabilities of the recurrent state transitions 0 → 0 and 1 → 1. The game stays in state 0 with probability 1 if the attacker selects action C and $l_t - a_t^{(1)} > 0$. Similarly, the game stays in state 1 with probability $1 - \phi_l$ if the attacker chooses action C and $l_t - a_t^{(1)} > 0$, where $\phi_{l_t} = 1/2l_t$ defines the probability that the intrusion is prevented.

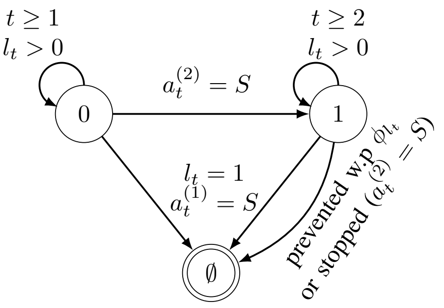

Fig. 3: State transition diagram of a game episode: each disk represents a state; an arrow represents a state transition; a label indicates the conditions for the state transition; a game episode starts in state $s_1 = 0$ with $l_1 = L$ and ends in state $s_T = \emptyset$.

Eq. 3 captures the transition 0 → 1, which occurs when the attacker chooses action S and $l_t - a_t^{(1)} > 0$. Eqs. 4-5 define the probabilities of the transitions to the terminal state $\emptyset$. The terminal state is reached in three cases: (*i*) when $l_t = 1$ and the defender takes the final stop action S (i.e. $l_t - a_t^{(1)} = 0$); (*ii*) when the intrusion is prevented with probability ϕ_{l_t}; and *(iii)*, when $s_t = 1$ and the attacker stops ($a_t^{(2)} = 1$).

The evolution of the game can be described with the state transition diagram in Fig. 3. The figure describes a game *episode*, which starts at $t = 1$ and ends at $t = T$. The time horizon T is a random variable that depends on both players' strategies and takes values in the set $\{t = 2, 3, \dots, \infty\}$.

Reward function $\mathcal{R}_l$: At each time-step t, the defender receives the reward $r_t = \mathcal{R}_l(s_t, (a_t^{(1)}, a_t^{(2)}))$ and the attacker receives the reward $-r_t$. The reward function is parameterized by the reward that the defender receives for stopping an intrusion ($R_{st} > 0$), the defender's cost of taking a defensive action ($R_{cost} < 0$), and its loss for being intruded ($R_{int} < 0$):

$$\mathcal{R}_l(\emptyset, \cdot) = 0, \qquad \mathcal{R}_l\big(1, (\cdot, S)\big) = 0 \tag{6}$$

$$\mathcal{R}_l\big(0, (C, \cdot)\big) = 0 \tag{7}$$

$$\mathcal{R}_l\big(0, (S, \cdot)\big) = R_{cost}/l \tag{8}$$

$$\mathcal{R}_l\big(1, (S, \cdot)\big) = R_{st}/l \tag{9}$$

$$\mathcal{R}_l\big(1, (C, C)\big) = R_{int} \tag{10}$$

Eq. 6 states that the reward is zero in the terminal state and when the attacker ends an intrusion. Eq. 7 states that the defender incurs no cost when it is not under attack and not taking defensive actions. Eq. 8 indicates that the defender incurs a cost when stopping if no intrusion is ongoing and Eq. 9 states that the defender receives a reward when taking a stop action that affects an ongoing intrusion. Lastly, Eq. 10 indicates that the defender receives a loss for each time-step that it is under intrusion.

Observation function $\mathcal{Z}$: At each time-step t, $o_t \in \mathcal{O}$ is drawn from a random variable O whose distribution f_O depends on the current state s_t. We define $\mathcal{Z}(o_t, s_t, (a^{(1)}_{t-1}, a^{(2)}_{t-1}))= \mathbb{P}[o_t|s_t, (a^{(1)}_{t-1}, a^{(2)}_{t-1})]$ as:

$$\mathcal{Z}\big(o_t, 0, \cdot\big) = f_O(o_t|0) \tag{11}$$

$$\mathcal{Z}\big(o_t, 1, \cdot\big) = f_O(o_t|1) \tag{12}$$

$$\mathcal{Z}\big(\emptyset, \emptyset, \cdot\big) = 1 \tag{13}$$

Belief update: At each time-step t, the belief state b_t is updated as follows:

$$b_{t+1}(s_{t+1}) = C \sum_{s_t \in \mathcal{S}} \sum_{a^{(2)}_t \in \mathcal{A}_2} \sum_{o_{t+1} \in \mathcal{O}} b(s_t) \pi_{2,l}(a^{(2)}_t | s_t, b_t) \cdot$$

$$\mathcal{Z}(o_{t+1}, s_{t+1}, (a^{(1)}_t, a^{(2)}_t)) \mathcal{T}\big(s_{t+1}, s_t, (a^{(1)}_t, a^{(2)}_t)\big) \tag{14}$$

where $C = 1/\mathbb{P}[o_{t+1}|a^{(1)}_1, \pi_{2,l}, b_t]$ is a normalizing factor that makes b_{t+1} sum to 1.

Player strategies $\pi_{i,l}$: A strategy of the defender is a function $\pi_{1,l} \in \Pi_1 : \mathcal{B} \rightarrow \Delta(\mathcal{A}_1)$. Analogously, a strategy of the attacker is a function $\pi_{2,l} \in \Pi_2 : \mathcal{S} \times \mathcal{B} \rightarrow \Delta(\mathcal{A}_2)$. $\Delta(\mathcal{A}_i)$ denotes the set of probability distributions over $\mathcal{A}_i$, Π_i denotes the strategy space of player i, and $\pi_{-i,l}$ denotes the strategy of player $j \in \mathcal{N} \setminus \{i\}$. For both players, a strategy is dependent on l but independent of t, i.e. strategies are stationary. If $\pi_{i,l}$ always maps on to an action with probability 1, it is called *pure*, otherwise it is called *mixed*.

Objective functions J_i: The goal of the defender is to *maximize* the expected discounted cumulative reward over the time horizon T. Similarly, the goal of the attacker is to *minimize* the same quantity. Therefore, the objective functions J_1 and J_2 are:

$$J_1(\pi_{1,l}, \pi_{2,l}) = \mathbb{E}_{(\pi_{1,l}, \pi_{2,l})}\left[\sum_{t=1}^{T} \gamma^{t-1} \mathcal{R}_l(s_t, \boldsymbol{a}_t)\right] \tag{15}$$

$$J_2(\pi_{1,l}, \pi_{2,l}) = -J_1(\pi_{1,l}, \pi_{2,l}) \tag{16}$$

where $\gamma \in [0, 1)$ is the discount factor.

Best response strategies $\tilde{\pi}_{i,l}$: A defender strategy $\tilde{\pi}_{1,l} \in B_1(\pi_{2,l})$ is called a *best response* against $\pi_{2,l} \in \Pi_2$ if it *maximizes* J_1 (Eq. 17). Similarly, an attacker strategy $\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$ is called a best response against $\pi_{1,l} \in \Pi_1$ if it *minimizes* J_1 (Eq. 18).

$$B_1(\pi_{2,l}) = \underset{\pi_{1,l} \in \Pi_1}{\arg\max}\, J_1(\pi_{1,l}, \pi_{2,l}) \tag{17}$$

$$B_2(\pi_{1,l}) = \underset{\pi_{2,l} \in \Pi_2}{\arg\min}\, J_1(\pi_{1,l}, \pi_{2,l}) \tag{18}$$

Optimal strategies $\pi^*_{i,l}$: An optimal defender strategy $\pi^*_{1,l}$ is a best response strategy against any attacker strategy that *minimizes* J_1. Similarly, an optimal attacker strategy $\pi^*_{2,l}$ is a best response against any defender strategy that *maximizes* J_1. Hence, when both players follow optimal strategies, they play best response strategies against each other:

$$(\pi^*_{1,l}, \pi^*_{2,l}) \in B_1(\pi^*_{2,l}) \times B_2(\pi^*_{1,l}) \tag{19}$$

This means that no player has an incentive to change its strategy and that $(\pi^*_{1,l}, \pi^*_{2,l})$ is a Nash equilibrium [32].

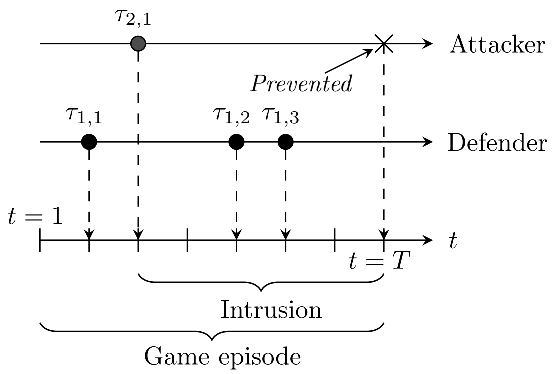

Fig. 4: Stopping times of the defender and the attacker in a game episode; the lower horizontal axis represents time; the black circles on the middle axis and the upper axis represent time-steps of defender stop actions and attacker stop actions, respectively; $\tau_{i,j}$ denotes the jth stopping time of player i; the cross shows the time the intrusion is prevented; an episode ends either when the attacker is prevented or when it takes its second stop action.

IV. Game-Theoretic Analysis and Our Approach for Finding Optimal Defender Strategies

Finding optimal strategies that satisfy Eq. 19 means finding a Nash equilibrium for the POSG Γ. We know from game theory that Γ has at least one mixed Nash equilibrium [33], [32], [34], [35]. (A Nash equilibrium is called mixed if one or more players follow mixed strategies.)

The equilibria of Γ can be obtained by finding pairs of strategies that are best responses against each other (Eq. 19). A best response for the defender is obtained by solving a POMDP $\mathcal{M}^P$, and a best response for the attacker is obtained by solving an MDP $\mathcal{M}$. Hence, the best response strategies can be expressed with Q-functions:

$$B_1(\pi_{2,l}) = \underset{a^{(1)}_t \in \mathcal{A}_1}{\arg\max}\, Q^*_{1,\pi_{2,l}}(b_t, a^{(1)}_t) \tag{20}$$

$$B_2(\pi_{1,l}) = \underset{a^{(2)}_t \in \mathcal{A}_2}{\arg\min}\, Q^*_{2,\pi_{1,l}}((b_t, s_t), a^{(2)}_t) \tag{21}$$

The corresponding Bellman equations are [36]:

$$Q^*_{i,\pi}(x_t, a^{(i)}_t) = \underset{\pi, x_t, a^{(i)}_t}{\mathbb{E}}\big[r_{t+1} + \gamma V^*_{i,\pi}(x_{t+1})\big] \tag{22}$$

$$V^*_{1,\pi}(x_t) = \max_{a^{(1)}_t \in \mathcal{A}_1} \underset{\pi, x_t, a^{(1)}_t}{\mathbb{E}}\big[r_{t+1} + \gamma V^*_{1,\pi}(x_{t+1})\big] \tag{23}$$

$$V^*_{2,\pi}(x_t) = \min_{a^{(2)}_t \in \mathcal{A}_2} \underset{\pi, a^{(2)}_t}{\mathbb{E}}\big[r_{t+1} + \gamma V^*_{2,\pi}(x_{t+1})\big] \tag{24}$$

$$V^*(b) = \max_{\pi_{1,l} \in \Delta(\mathcal{A}_1)} \min_{\pi_{2,l} \in \Delta(\mathcal{A}_2)} \mathbb{E}_{\pi_{1,l}, \pi_{2,l}, b}\big[r^{(1)}_{t+1} + \gamma V^*(b_{t+1})\big] \tag{25}$$

Since the game is zero-sum, stationary, and $\gamma < 1$, it follows from game theory that $V^*(b) = V^*_{1,\pi^*_{2,l}}(b) = V^*_{2,\pi^*_{1,l}}(b, s)$ [33], [37]. Further, from Markov decision theory we know that for any strategy pair $(\pi_{1,l}, \pi_{2,l})$, a corresponding pair of best response strategies $(\tilde{\pi}_{1,l} \in B_1(\pi_{2,l}), \tilde{\pi}_{2,l} \in B_2(\pi_{1,l}))$ exists [38].

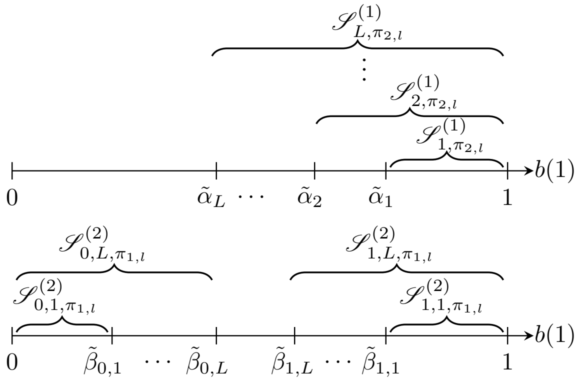
$\mathscr{S}^{(1)}_{L,\pi_{2,l}}$
$\vdots$
$\mathscr{S}^{(1)}_{2,\pi_{2,l}}$
$\mathscr{S}^{(1)}_{1,\pi_{2,l}}$
$b(1)$
0
$\tilde{\alpha}_L \cdots$
$\tilde{\alpha}_2$
$\tilde{\alpha}_1$
1
$\mathscr{S}^{(2)}_{0,L,\pi_{1,l}}$
$\mathscr{S}^{(2)}_{1,L,\pi_{1,l}}$
$\mathscr{S}^{(2)}_{0,1,\pi_{1,l}}$
$\mathscr{S}^{(2)}_{1,1,\pi_{1,l}}$
$b(1)$
0
$\tilde{\beta}_{0,1}$
$\cdots$ $\tilde{\beta}_{0,L}$
$\tilde{\beta}_{1,L}$ $\cdots$ $\tilde{\beta}_{1,1}$
1

Fig. 5: Illustration of Theorem 1; the upper plot shows the existence of L thresholds $\tilde{\alpha}_1 \geq \tilde{\alpha}_2, \dots, \geq \tilde{\alpha}_L \in [0,1]$ that define a best response strategy $\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}} \in B_1(\pi_{2,l})$ (Eq. 26); the lower plot shows the existence of $2L$ thresholds $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \dots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$ that define a best response strategy $\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}} \in B_2(\pi_{1,l})$ (Eqs. 27-28).

A. Analyzing Best Responses using Optimal Stopping Theory

The POMDP $\mathcal{M}^P$ and the MDP $\mathcal{M}$ that determine the best response strategies can be understood as *optimal stopping* problems (see Fig. 4) [14], [39], [40], [11]. In the defender's case, the problem is to find a stopping strategy $\pi^*_{1,l}(b_t) \rightarrow \{S, C\}$ that maximizes J_1 (Eq. 15) and prescribes the optimal stopping times $\tau^*_{1,1}, \tau^*_{1,2}, \dots, \tau^*_{1,L}$. Similarly, the problem for the attacker is to find a stopping strategy $\pi^*_{2,l}(s_t, b_t) \rightarrow \{S, C\}$ that minimizes J_1 (Eq. 16) and prescribes the optimal stopping times $\tau^*_{2,1}$ and $\tau^*_{2,2}$.

Given a pair of stopping strategies $(\pi_{1,l}, \pi_{2,l})$ and their best responses $(\tilde{\pi}_{1,l} \in B_1(\tilde{\pi}_{2,l}), \tilde{\pi}_{2,l} \in B_2(\tilde{\pi}_{1,l}))$, we define two subsets of $\mathcal{B}$: the *stopping sets* and the *continuation sets*.

The stopping sets contain the belief states where S is a best response: $\mathscr{S}^{(1)}_{l,\pi_{2,l}} = \{b(1) \in [0,1] : \tilde{\pi}_{1,l}\big(b(1)\big) = S\}$ and $\mathscr{S}^{(2)}_{s,l,\pi_{1,l}} = \{b(1) \in [0,1] : \tilde{\pi}_{2,l}\big(s, b(1)\big) = S\}$. Similarly, the continuation sets contain the belief states where C is a best response: $\mathscr{C}^{(1)}_{l,\pi_{2,l}} = \{b(1) \in [0,1] : \tilde{\pi}_{1,l}\big(b(1)\big) = C\}$ and $\mathscr{C}^{(2)}_{s,l,\pi_{1,l}} = \{b(1) \in [0,1] : \tilde{\pi}_{2,l}\big(s, b(1)\big) = C\}$.

Based on [41], [42], [43], [11], [37], we formulate Theorem 1, which contains an existence result for equilibria and a structural result for best response strategies in the game.

Theorem 1. *Given the one-sided POSG Γ in Section III with $L \geq 1$, the following holds.*

(A) Γ has a mixed Nash equilibrium. Further, Γ has a pure Nash equilibrium when $s = 0 \iff b(1) = 0$.

(B) Given any attacker strategy $\pi_{2,l}$, if the probability mass function $f_{O|s}$ is totally positive of order 2 (i.e., TP2 [41, Definition 10.2.1, pp. 223]), there exist values $\tilde{\alpha}_1 \geq \tilde{\alpha}_2 \geq \dots \geq \tilde{\alpha}_L \in [0,1]$ and a best response strategy $\tilde{\pi}_{1,l} \in B_1(\pi_{2,l})$ of the defender that satisfies:

$$\tilde{\pi}_{1,l}(b(1)) = S \iff b(1) \geq \tilde{\alpha}_l \qquad l \in 1,\dots,L \tag{26}$$

(C) Given a defender strategy $\pi_{1,l}$, where $\pi_{1,l}(S|b(1))$ is non-decreasing in $b(1)$ and $\pi_{1,l}(S|1) = 1$, there exist values $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \dots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$ and a best response strategy $\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$ of the attacker that satisfies:

$$\tilde{\pi}_{2,l}(0, b(1)) = C \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{0,l} \tag{27}$$

$$\tilde{\pi}_{2,l}(1, b(1)) = S \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{1,l} \tag{28}$$

for $l \in 1, \dots, L$.

Proof. See Appendices A-C. □

Theorem 1 tells us that Γ has a mixed Nash equilibrium. It also tells us that, under certain assumptions, the best response strategies have threshold properties. In the following, we describe an efficient algorithm that takes advantage of these properties to approximate equilibria of Γ.

B. Finding Nash Equilibria through Fictitious Self-Play

Computing Nash equilibria for a POSG is generally intractable [37]. However, approximate solutions can be obtained through iterative approximation methods. One such method is *fictitious self-play*, where both players start from random strategies and continuously update their strategies based on the outcomes of played game episodes [44].

Fictitious self-play evolves through a sequence of iteration steps, which is illustrated in Fig. 6. An iteration step includes three procedures. First, player 1 learns a best response strategy against player 2's current strategy. After that, the roles are reversed and player 2 learns a best response strategy against player 1's current strategy. Next, the iteration step is completed by having each player adopt a new strategy. The new strategies are determined by the empirical distributions over the past best response strategies. The sequence of iteration steps continues until the strategies of both players have sufficiently converged to a Nash equilibrium [44], [45].

C. Our Self-Play Algorithm: T-FP

We present a fictitious self-play algorithm, which we call T-FP, that exploits the statements in Theorem 1 to efficiently approximate Nash equilibria of Γ.

T-FP implements the fictitious self-play process described in Section IV-B and generates a sequence of strategy profiles $(\pi_{1,l}, \pi_{2,l})$, $(\pi'_{1,l},\ \pi'_{2,l})$, $\dots$, $(\pi^*_{1,l}, \pi^*_{2,l})$ that converges to a Nash equilibrium. During each step of this process, T-FP learns best responses against the players' current strategies and then updates the strategies of both players to be the empirical distribution over the past strategies (see Fig. 6).

T-FP parameterizes the best response strategies $\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}} \in B_1(\pi_{2,l})$ and $\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}} \in B_2(\pi_{1,l})$ by threshold vectors. The defender's best response strategy is parameterized with the vector $\tilde{\theta}^{(1)} \in \mathbb{R}^L$ (Eq. 30). Similarly, the attacker's best response strategy is parameterized with the vector $\tilde{\theta}^{(2)} \in \mathbb{R}^{2L}$ (Eq. 31).

$$\varphi(a,b) = \left(1 + \left(\frac{b(1-\sigma(a))}{\sigma(a)(1-b)}\right)^{-20}\right)^{-1} \tag{29}$$

$$\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}\big(S|b(1)\big) = \varphi\left(\tilde{\theta}^{(1)}_l, b(1)\right) \tag{30}$$

$$\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}}\big(S|b(1), s\big) = \varphi\left(\tilde{\theta}^{(2)}_{sL+l}, \pi_{1,l}(S|b(1))\right) \tag{31}$$

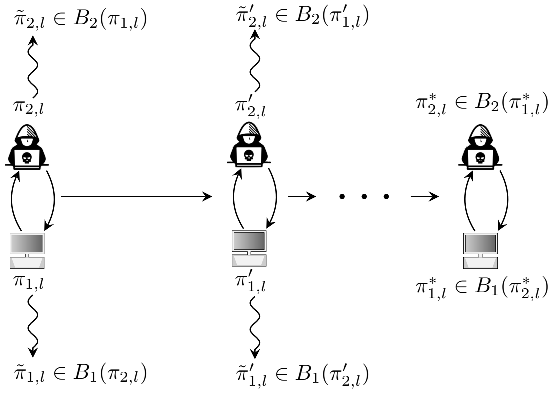
$\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$
$\pi_{2,l}$
$\pi_{1,l}$
$\tilde{\pi}_{1,l} \in B_1(\pi_{2,l})$
$\tilde{\pi}'_{2,l} \in B_2(\pi'_{1,l})$
$\pi'_{2,l}$
$\pi'_{1,l}$
$\tilde{\pi}'_{1,l} \in B_1(\pi'_{2,l})$
$\pi^*_{2,l} \in B_2(\pi^*_{1,l})$
$\pi^*_{1,l} \in B_1(\pi^*_{2,l})$

Fig. 6: The fictitious self-play process; in each iteration both players learn a best response strategy $\tilde{\pi}_{i,l} \in B_i(\pi_{-i,l})$ and update their strategies based on the empirical distribution of past best responses; the horizontal arrows indicate the iterations of self-play and the vertical arrows indicate the learning of best responses; if the process is convergent, it reaches a Nash equilibrium $(\pi^*_{1,l}, \pi^*_{2,l})$.

$\sigma(\cdot)$ is the sigmoid function, $\sigma(\tilde{\theta}^{(1)}_1)$, $\sigma(\tilde{\theta}^{(1)}_2)$, ..., $\sigma(\tilde{\theta}^{(1)}_L) \in [0,1]$ are the L thresholds of the defender (see Theorem 1.B), and $\sigma(\tilde{\theta}^{(2)}_1)$, $\sigma(\tilde{\theta}^{(2)}_2)$, ..., $\sigma(\tilde{\theta}^{(2)}_{2L}) \in [0,1]$ are the $2L$ thresholds of the attacker (see Theorem 1.C).

Using this parameterization, T-FP learns the best response strategies by iteratively updating $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic approximation. To update the threshold vectors, T-FP uses simulations of Γ, which allows to evaluate the objective functions $J_1(\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}, \pi_{2,l})$ (Eq. 15) and $J_2(\pi_{1,l}, \tilde{\pi}_{2,l,\tilde{\theta}^{(2)}})$ (Eq. 16). The obtained values of J_1 and J_2 are then used to estimate the gradients $\nabla_{\tilde{\theta}^{(1)}} J_1$ and $\nabla_{\tilde{\theta}^{(1)}} J_2$ using the Simultaneous Perturbation Stochastic Approximation (SPSA) gradient estimator [46], [47]. Next, the estimated gradients are then used to update $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic gradient ascent. This procedure continues until $\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}$ and $\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}}$ have sufficiently converged.

After the process of learning best response strategies has converged, the threshold vectors are added to buffers $\Theta^{(1)}$ and $\Theta^{(2)}$, which contain the vectors learned in previous iterations of T-FP. The iteration step is completed by having both players update their strategies based on the empirical distributions over the past vectors in the buffer.

The pseudocode of T-FP is listed in Algorithm 1. (Here $\mathcal{U}_k(\{-1,1\})$ denotes a k-dimensional discrete multivariate uniform distribution on $\{-1,1\}$.)

V. Emulating the Target Infrastructure to Instantiate the Simulation

To simulate a game episode we must know the observation distribution conditioned on the system state (see Eqs. 11-13). We estimate this distribution using measurements from the emulation system shown in Fig. 2. Moreover, to evaluate the performance of strategies learned in the simulation system, we run game episodes in the emulation system having the attacker and the defender take actions at the times prescribed by the learned strategies.

Algorithm 1 T-FP

```
Input
$\Gamma, N$: the POSG and # best response iterations
$a, c, \lambda, A, \epsilon, \delta$: scalar coefficients
Output
$(\pi^*_{1,l}, \pi^*_{2,l})$: an approximate Nash equilibrium
procedure T-FP
    $\tilde{\theta}^{(1)} \sim \mathcal{U}_L(\{-1,1\}), \quad \tilde{\theta}^{(2)} \sim \mathcal{U}_{2L}(\{-1,1\})$
    $\Theta^{(1)} \leftarrow \{\tilde{\theta}^{(1)}\}, \quad \Theta^{(2)} \leftarrow \{\tilde{\theta}^{(2)}\}, \quad \hat{\delta} \leftarrow \infty$
    $\pi_{1,l} \leftarrow \textsc{EmpiricalDistribution}(\Theta^{(1)})$
    $\pi_{2,l} \leftarrow \textsc{EmpiricalDistribution}(\Theta^{(2)})$
    while $\hat{\delta} \geq \delta$ do
        for $i \in \{1,2\}$ do
            $\tilde{\theta}^{(i)}_{(1)} \sim \mathcal{U}_{iL}(\{-1,1\})$
            for $n \in \{1,\dots,N\}$ do
                $a_n \leftarrow \frac{a}{(n+A)^{\epsilon}}, \quad c_n \leftarrow \frac{c}{n^{\lambda}}$
                for $k \in \{1,\dots,iL$ do
                    $(\Delta_n)_k \sim \mathcal{U}_1(\{-1,1\})$
                end for
                $R_{high} \sim J_i(\pi_{i,l,\tilde{\theta}^{(i)}_{(n)}} + c_n\Delta_n, \pi_{-i,l})$
                $R_{low} \sim J_i(\pi_{i,l,\tilde{\theta}^{(i)}_{(n)}} - c_n\Delta_n, \pi_{-i,l})$
                for $k \in \{1,\dots,iL\}$ do
                    $G \leftarrow \frac{R_{high}-R_{low}}{2c_n(\Delta_n)_k}$
                    $\left(\hat{\nabla}_{\tilde{\theta}^{(i)}_{(n)}} J_i(\pi_{i,l,\tilde{\theta}^{(i)}_{(n)}}, \pi_{-i,l})\right)_k \leftarrow G$
                end for
                $\tilde{\theta}^{(i)}_{(n+1)} = \tilde{\theta}^{(i)}_{(n)} + a_n \hat{\nabla}_{\tilde{\theta}^{(i)}_{(n)}} J_i(\pi_{i,l,\tilde{\theta}^{(i)}_{(n)}}, \pi_{-i,l})$
            end for
            $\Theta^{(i)} \leftarrow \Theta^{(i)} \cup \tilde{\theta}^{(i)}_{(N+1)}$
        end for
        $\pi_{1,l} \leftarrow \textsc{EmpiricalDistribution}(\Theta^{(1)})$
        $\pi_{2,l} \leftarrow \textsc{EmpiricalDistribution}(\Theta^{(2)})$
        $\hat{\delta} = \textsc{Exploitability}(\pi_{1,l}, \pi_{2,l})$
    end while
    return $(\pi_{1,l}, \pi_{2,l})$
end procedure
```

A. Emulating the Target Infrastructure

The emulation system executes on a cluster of machines that runs a virtualization layer provided by Docker [48] containers and virtual links. The system implements network isolation and traffic shaping on the containers using network namespaces and the NetEm module in the Linux kernel [49]. Resource constraints on the containers, e.g. CPU and memory constraints, are enforced using cgroups.

The network topology of the emulated infrastructure is given in Fig. 1 and the configuration is given in Appendix E. The system emulates the clients, the attacker, the defender, 31 physical components of the target infrastructure (e.g application servers and the gateway), as well as network connectivity. The software functions replicate important components of the target infrastructure, such as, web servers, databases, and the Snort IPS, which is deployed using Snort's community ruleset v2.9.17.1.

We emulate connections between servers as full-duplex loss

Functions	*Application servers*
HTTP, SSH, SNMP, ICMP	N_2, N_3, N_{10}, N_{12}
IRC, PostgreSQL, SNMP	$N_{31}, N_{13}, N_{14}, N_{15}, N_{16}$
FTP, DNS, Telnet	N_{10}, N_{22}, N_4

TABLE 1: Emulated client population; each client interacts with application servers using a set of network functions.

Stop index	*Action*
1	Revoke user certificates
2	Blacklist IPs
3	Drop traffic that generates IPS alerts of priority 1
4	Drop traffic that generates IPS alerts of priority 2
5	Drop traffic that generates IPS alerts of priority 3
6	Drop traffic that generates IPS alerts of priority 4
7	Block gateway

TABLE 2: Defender stop actions in the emulation.

less connections with capacity 1 Gbit/s in both directions. We emulate external connections between the gateway and the client population as full-duplex connections of 100 Mbit/s capacity and 0.1% packet loss with random bursts of 1% packet loss. (These numbers are drawn from empirical studies on enterprise and wide area networks [50], [51], [52].)

B. Emulating the Client Population

The *client population* is emulated by processes that run inside Docker containers and interact with the application servers through the gateway. The clients select functions uniformly at random from the list given in Table 1. We emulate client arrivals using a stationary Poisson process with parameter $\lambda = 20$ and exponentially distributed service times with parameter $\mu = \frac{1}{4}$. The duration of a time-step in the emulation is 30 seconds.

C. Emulating Defender and Attacker Actions

The attacker and the defender observe the infrastructure continuously and take actions at discrete time-steps $t = 1, 2, \ldots, T$. During each step, the defender and the attacker can perform one action each.

The defender executes either a continue action or a stop action. Only the stop action affects the progression of the emulation. We have implemented $L = 7$ stop actions, which are listed in Table 2. The first stop action revokes user certificates and recovers user accounts thought to be compromised by the attacker. The second stop action updates the firewall configuration of the gateway to drop traffic from IP addresses that have been flagged by the IPS. Stop actions $3 - 6$ update the configuration of the IPS to drop traffic that generates alerts of priorities $1 - 4$. The final stop action blocks all incoming traffic. (Contrary to Snort's terminology, we define 4 to be the highest priority.)

Like the defender, the attacker executes either a stop action or a continue action during each time-step. The attacker can take two stop actions. The first determines when the intrusion starts and the second determines when it ends (see Section III-A). A continue action in state $s = 0$ has no affect on the emulation, but a continue action in state $s = 1$ has. When the attacker takes a stop action in state $s = 0$ or a continue action in state $s = 1$, an intrusion command is executed. We have implemented 25 such commands, which are listed in Table 3. During each step of an intrusion, the attacker selects a command uniformly at random from the list in Table 3.

Type	*Actions*
Reconnaissance	TCP-SYN scan, UDP port scan, TCP Null scan, TCP Xmas scan, TCP FIN scan, ping-scan, TCP connection scan, "Vulscan" vulnerability scanner
Brute-force attack	Telnet, SSH, FTP, Cassandra, IRC, MongoDB, MySQL, SMTP, Postgres
Exploit	CVE-2017-7494, CVE-2015-3306, CVE-2010-0426, CVE-2015-5602, CVE-2014-6271, CVE-2016-10033 CVE-2015-1427, SQL Injection

TABLE 3: Attacker commands to emulate intrusions.

D. Estimating the IPS Alert Distribution

At the end of every time-step, the emulation system collects the metric o_t, which contains the number of IPS alerts that occurred during the time-step, weighted by priority. For the evaluation reported in this paper we collect measurements from 23000 time-steps of 30 seconds each.

Using these measurements, we fit a Gaussian mixture distribution $\hat{f}_O$ as an estimate of f_O in the target infrastructure (Eqs. 11-12). For each state s, we obtain the conditional distribution $\hat{f}_{O|s}$ through expectation-maximization [53].

Fig. 7 shows the empirical distributions and the fitted model over the discrete observation space $\mathcal{O} = \{1, 2, \ldots, 9000\}$. $\hat{f}_{O|0}$ and $\hat{f}_{O|1}$ are Gaussian mixtures with two and three components, respectively. Both mixtures have most probability mass within the range $0 - 1000$. $\hat{f}_{O|1}$ also has substantial probability mass at larger values.

The stochastic matrix with the rows $\hat{f}_{O|0}$ and $\hat{f}_{O|1}$ has about 72×10^6 minors, out of which virtually all are non-negative. This suggests to us that the TP2 assumption in Theorem 1 can be made.

E. Running Game Episodes

During a simulation, the game state evolves according to the dynamics described in Section III. The actions of both players are determined by their strategies, and the observations are sampled from the estimated observation distribution $\hat{f}_O$. The defender's belief evolve according to Eq. 14.

An episode in the emulation system differs from an episode in the simulation system. First, the emulated client population issues requests to the emulated application servers (see Section V-B). Second, the defender's observations are not sampled but are obtained through reading log files and metrics of the emulated infrastructure, which depend on the network traffic

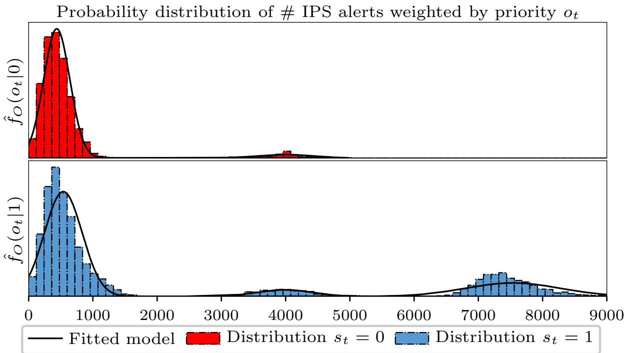

Fig. 7: Empirical distributions of o_t when no intrusion occurs ($s_t = 0$) and during intrusion ($s_t = 1$); the black lines show the fitted Gaussian mixture models.

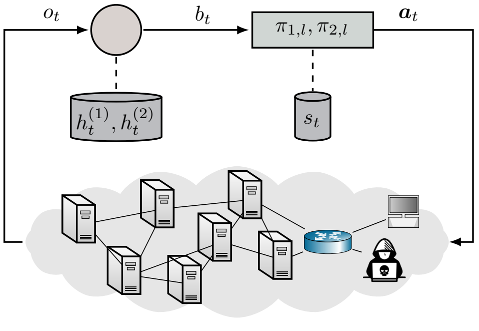

Fig. 8: Emulation of a game episode; measurement data (o_t) is aggregated in a log that is consumed by a stream processor to compute the next belief b_t based on the history h_t; the next pair of actions $\boldsymbol{a}_t$ is sampled from the strategy pair $(\pi_{1,l}, \pi_{2,l})$ and is executed in the infrastructure using a gRPC/SSH API.

generated by the client population, the attacker, as well as internal infrastructure processes. Third, attacker and defender actions in the emulation system include executing computing and networking functions (see Table 2 and Table 3), which are not captured in the game simulation.

We collect the observations in the emulation system using a distributed log implemented with Kafka [54] (see Fig. 8). Log updates are read periodically by a program that computes the defender's belief state $b(1)$ and the game state s. Using this information, the attacker's and the defender's strategies determine the next actions, which are executed in the emulation system using an API implemented over gRPC [55] and SSH.

VI. Learning Nash Equilibrium Strategies for the Target Infrastructure

Our approach to finding effective defender strategies includes: (1) extensive simulation of game episodes in the simulation system to learn Nash equilibrium strategies; and (2) evaluation of the learned strategies on the emulation system (see Fig. 2). This section describes our evaluation results for the intrusion prevention use case.

The environment for running simulations and training strategies is a Tesla P100 GPU. The hyperparameters for the training algorithm are listed in Appendix D. The emulated infrastructure is deployed on a server with a 24-core Intel Xeon Gold 2.10 GHz CPU and 768 GB RAM.

The code for the simulation system and the measurement traces for the intrusion prevention use case are available at [56]. They can be used to validate our results and extend this research.

A. *Learning Equilibrium Strategies through Self-Play*

We run T-FP for 500 iterations to estimate a Nash equilibrium using the iterative method described in Section IV-B. At the end of each iteration step, we evaluate the current strategy pair $(\pi_{1,l}, \pi_{2,l})$ by running 500 evaluation episodes in the simulation system and 5 evaluation episodes in the emulation system. This allows us to produce learning curves for different performance metrics (see Fig. 9).

To estimate the convergence of the sequence of strategy pairs to a Nash equilibrium, we use the *approximate exploitability* metric v^{exp} [57]:

$$v^{exp} = J_1(\hat{\pi}_{1,l}, \pi_{2,l}) + J_2(\pi_{1,l}, \hat{\pi}_{2,l}) \qquad (32)$$

where $\hat{\pi}_{i,l}$ denotes an approximate best response strategy for player i obtained through dynamic programming. The closer v^{exp} becomes to 0, the closer $(\pi_{1,l}, \pi_{2,l})$ is to a Nash equilibrium.

The 500 training iterations constitute one *training run*. We run four training runs with different random seeds. A single training run takes about 5 hours of processing time on a P100 GPU. In addition, it takes around 12 hours to evaluate the strategies on the emulation system.

Defender baseline strategies: When we evaluate the learned defender strategies, we compare them with three baselines. The first baseline prescribes the stop action whenever an IPS alert occurs, i.e., whenever $o_t \geq 1$. The second baseline follows the Snort IPS's internal recommendation system and takes a stop action whenever 100 IP packets have been dropped by the Snort IPS (see Appendix E for the Snort configuration). The third baseline assumes knowledge of the exact intrusion time and performs all stop actions at subsequent time-steps.

Baseline algorithms: We compare the performance of T-FP with two baseline algorithms: Neural Fictitious Self-Play (NFSP) [58] and Heuristic Search Value Iteration (HSVI) for one-sided POSGs [59]. NFSP is a state-of-the-art deep reinforcement learning algorithm for imperfect-information games. Contrary to T-FP, NFSP does not exploit the threshold structures expressed in Theorem 1 and as a result is more complex. HSVI is a state-of-the-art dynamic programming algorithm for one-sided POSGs.

B. *Discussion of the Evaluation Results*

Fig. 9 shows the learning curves of the strategies obtained during the self-play process. The red curve represents the results from the simulation system and the blue curves show the results from the emulation system. The purple and orange curves give the performance of the Snort IPS baseline and the baseline strategy that mandates a stop action whenever an IPS alert occurs, respectively. The dashed black curve gives the

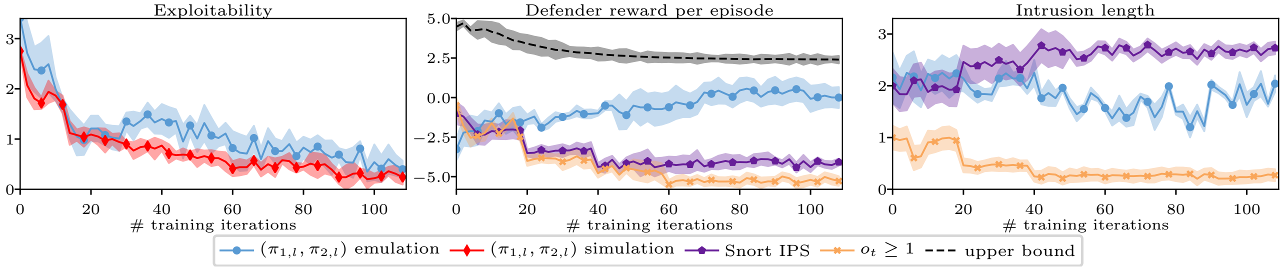

Fig. 9: Learning curves from the self-play process with T-FP; the red curve show simulation results and the blue curves show emulation results; the purple, orange, and black curves relate to baseline strategies; the figures show different performance metrics: exploitability, episodic reward, and the length of intrusion; the curves indicate the mean and the 95% confidence interval over four training runs with different random seeds.

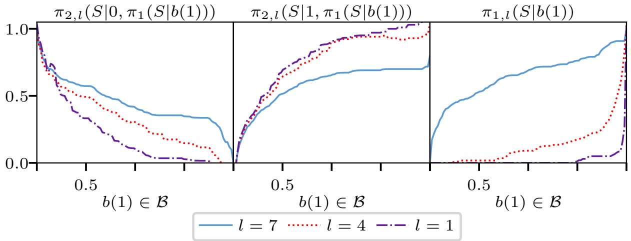

Fig. 10: Probability of the stop action S by the learned equilibrium strategies in function of $b(1)$ and l; the left and middle plots show the attacker's stopping probability when $s = 0$ and $s = 1$, respectively; the right plot shows the defender's stopping probability.

performance of the baseline strategy that assumes knowledge of the exact intrusion time.

The results in Fig. 9 lead us to the following conclusions. First, the fact that all learning curves seem to converge suggests to us that the learned strategies have converged as well. Second, we observe that the exploitability of the learned strategies converges to small values (left plot of Fig. 9). This indicates that the learned strategies approximate Nash equilibria both in the simulation system and in the emulation system. Third, we see from the middle plot in Fig. 9 that both baseline strategies show decreasing performance as the attacker updates its strategy. In contrast, the learned defender strategy improves its performance over time. This shows the benefit of using a game-theoretic approach, whereby the defender's strategy is optimized against a dynamic attacker.

Fig. 10 illustrates some of the structural properties of the learned strategies. The y-axis shows the probability of the stop action S and the x-axis shows the defender's belief $b(1) \in \mathcal{B}$. We observe that the strategies are stochastic. Hence, since Fig. 9 suggests that the learned strategies converge to a Nash equilibrium, Fig. 10 suggests that this equilibrium is *mixed*, which we expect based on Theorem 1.A. As we further expect from Theorem 1.B-C, we see that that the defender's stopping probability is increasing with $b(1)$ and decreasing with l (right plot of Fig. 10). Similarly, we observe that the attacker's stopping probability is decreasing with the defender's stopping probability when $s = 0$ and is increasing when $s = 1$ (left and middle plot of Fig. 10).

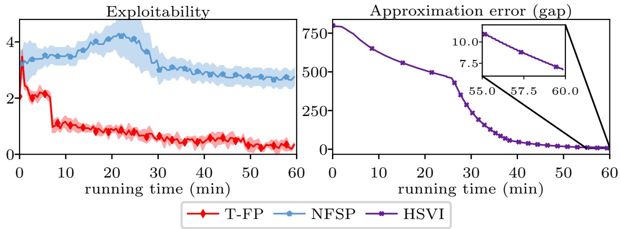

Fig. 11: Comparison between T-FP and two baseline algorithms: NFSP and HSVI; all curves show simulation results; the red curve relate to T-FP; the blue curve to NFSP; the purple curve to HSVI; the left plot shows the approximate exploitability metric and the right plot shows the HSVI approximation error [59]; the curves depicting T-FP and NFSP show the mean and the 95% confidence interval over four training runs with different random seeds.

Fig. 11 allows a comparison between T-FP and the two baseline algorithms (NFSP and HSVI) as they execute in the simulation system. Since T-FP and NFSP both implement fictitious self-play, they allow for a direct comparison. We observe that T-FP converges much faster to a Nash equilibrium than NFSP. We expect the fast convergence of T-FP due to its design to exploit structural properties of the stopping game.

The right plot of Fig. 11 shows that HSVI reaches an HSVI approximation error < 5 within an hour. We expected slower convergence due to findings in [60], [37]. A direct comparison between T-FP and HSVI is not possible due to the different nature of the two algorithms.

Lastly, Fig. 12 shows the computed value function of the game $\hat{V}^*$ (Eq. 25). We see that $\hat{V}^*$ is piece-wise linear and convex, as expected by the theory of one-sided POSGs [60]. We also observe that the value of $\hat{V}^*$ is minimal when $b(1) \approx 0.25$ and is 0 when $b(1) = 1$. Moreover, we note that this value is slightly lower for $l = 1$ than for $l = 7$.

VII. Related Work

Traditional approaches to intrusion prevention use packet inspection and static rules for detection of intrusions and selection of response actions [61], [62], [1]. Their main drawback

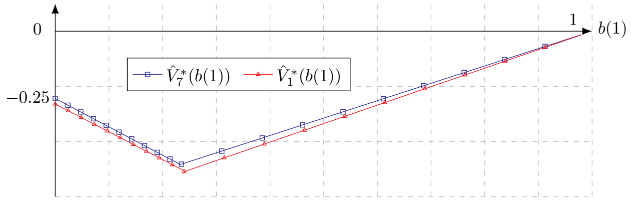

Fig. 12: The value function $\hat{V^*}_l(b(1))$ computed through the HSVI algorithm with approximation error 4; the blue and red curves relate to $l = 7$ and $l = 1$, respectively.

lies in the need for domain experts to configure the rule sets. As a consequence, much effort has been devoted to developing methods for finding security strategies in an automatic way. This research uses concepts and methods from various areas, most notably from anomaly detection (see example [63]), change-point detection (see example [64]), statistical learning (see examples [65], [66], [67]), control theory (see survey [5]), game theory (see textbooks [9], [68], [69], [70]), artificial intelligence (see survey [71]), dynamic programming (see example [4]), reinforcement learning (see surveys [72], [73]), evolutionary methods (see example [6]), and attack graphs (see example [74]). While the research reported in this paper is informed by all the above works, we limit the following discussion to prior work that uses game-theoretic models and centers around finding strategies through reinforcement learning.

A. Reinforcement Learning in Network Security

Many recent results of automating security strategies have been obtained using reinforcement learning methods. In particular, a large number of studies have focused on intrusion prevention use cases similar to the one we discuss in this paper [12], [13], [25], [26], [18], [24], [27], [19], [28], [20], [21], [23], [75], [76], [77], [22], [78].

These works use a variety of models, including MDPs [18], [19], [20], [21], [22], Markov games [25], [12], [75], and POMDPs [13], [23], [24], as well as various reinforcement learning algorithms, including Q-learning [25], [18], [19], SARSA [24], PPO [12], [13], hierarchical reinforcement learning [20], DQN [21], Thompson sampling [23], MuZero [75], NFQ [76], DDQN [22], and DDPG [77].

This paper differs from the works referenced above in three main ways. First, we model the intrusion prevention use case as a partially observed stochastic game. Most of the other works model the use case as a single-agent MDP or POMDP. The advantage of using a game-theoretic model is that it allows finding defender strategies that are effective against dynamic attackers.

Second, we derive structural properties of strategies in the game using optimal stopping theory. This type of analysis is not provided in the referenced works.

Third, our method to find effective defender strategies includes using an emulation system in addition to a simulation system. The advantage of our method compared to the simulation-only approaches [12], [13], [25], [26], [18], [24], [27], [19], [28], [20], [21], [23] is that the parameters of our simulation system are determined by measurements from an emulation system instead of being chosen by a human expert. Further, the learned strategies are evaluated in the emulation system, not in the simulation system. As a consequence, the evaluation results give higher confidence of the obtained strategies' performance in the target infrastructure than what simulation results would provide.

Some prior work on automated intrusion prevention that make use of emulation are: [75], [76], [77], [79], and [22]. They emulate software-defined networks based on Mininet [80]. The main differences between these efforts and the work described in this paper are: (1) we develop our own emulation system which allows for experiments with a large variety of exploits; (2) we focus on a different intrusion prevention use case; (3) we do not assume that the defender has perfect observability; and (4), we use an underlying theoretical framework to formalize the use case, derive structural properties of optimal strategies, and test these properties in an emulation system.

Finally, [81] and [82] describe ongoing efforts in building emulation platforms for reinforcement learning, which resemble our emulation system. In contrast to these papers, our emulation system has been built to investigate the specific use case of intrusion prevention and forms an integral part of our general solution method (see Fig. 2).

B. Game Theoretic Modeling in Network Security

Several examples of game theoretic security models can be found in the literature, e.g. advanced persistent threat games [29], [30], [31], honeypot placement games [83], [84], [85], resource allocation games [86], authentication games [10], distributed denial-of-service games [79], [87], and intrusion prevention games [88], [12], [75], [89], [90], [91].

This paper differs from the works referenced above in two main ways. First, we model the intrusion prevention use case as an optimal stopping game. The benefit of our model is that it provides insight into the structure of best response strategies through the theory of optimal stopping.

Game-theoretic formulations based on optimal stopping theory can be found in prior research on Dynkin games [15], [92], [93], [94], [95]. Compared to these papers, our approach is more general by (1) allowing each player to take multiple stop actions within an episode; and (2), by not assuming a game of perfect information. Another difference is that the referenced papers either study purely mathematical problems or problems in mathematical finance. To the best of our knowledge, we are the first to apply the stopping game formulation to the use case of intrusion prevention.

Our stopping game has similarities with the FlipIt game [29] and signaling games [96], both of which are commonplace in the security literature (see survey [97] and textbooks [9], [68], [69], [70]). Signaling games have the same information asymmetry as our game and FlipIt uses the same binary state space to model the state of an intrusion. The main differences are as follows. FlipIt models the use case of advanced persistent threats and is a symmetric non-zero-sum game. In contrast, our game models an intrusion prevention use case and is an asymmetric zero-sum game. Compared to signaling games, the main differences are that our game is a sequential and simultaneous-move game. Signaling games are typically two-stage games where one player moves in each stage.

Second, as we noted above, we evaluate the obtained strategies on an emulated IT infrastructure. This contrasts with most of the prior works that use game-theoretic approaches, which only evaluate strategies in simulation [29], [30], [31], [83], [84], [85], [10], [87], [88], [12], [89], [90], [91].

VIII. Conclusion and Future Work

We formulate the interaction between an attacker and a defender in an intrusion prevention use case as an optimal stopping game. The theory of optimal stopping provides us with insight about optimal strategies for attackers and defenders. Based on this knowledge, we develop a fictitious self-play algorithm, T-FP, which allows us to compute near optimal strategies in an efficient way. This approach provides us with a complete formal framework for analyzing and solving the intrusion prevention use case. The simulation results from executions of T-FP show that the exploitability of the computed strategies converges, which suggests that the strategies converge to a Nash equilibrium and thus to an optimum in the game-theoretic sense. The results also demonstrate that T-FP converges faster than a state-of-the-art fictitious self-play algorithm by taking advantage of structural properties of optimal stopping strategies.

To assess the computed strategies in a real environment, we evaluate them in a system that emulates our target infrastructure. The results show that the strategies achieve almost the same performance in the emulated infrastructure as in the simulation, which gives us a high confidence of their performance in the target infrastructure.

We plan to extend this work in several directions. First of all, the model of the attacker and the defender in this paper is simplistic as it only models the timing of actions and not their selection. We plan to combine our current model for deciding when to take defensive actions with a model for the selection of which action to take.

IX. Acknowledgments

This research has been supported in part by the Swedish armed forces and was conducted at KTH Center for Cyber Defense and Information Security (CDIS). The authors would like to thank Pontus Johnson for his useful input to this research, and Forough Shahab Samani and Xiaoxuan Wang for their constructive comments on a draft of this paper. The authors are also grateful to Branislav Bosanský for sharing the code of the HSVI algorithm for one-sided POSGs and to Jakob Stymne for contributing to our implementation of NFSP.

Appendix A
Proof of Theorem 1.A

Proof of Theorem 1.A. Since the POSG Γ introduced in Section III is finite and $\gamma \in (0,1)$, the existence proofs in [35] and [60] applies, which state that a mixed Nash equilibrium exists.

We prove that a pure Nash equilibrium exists when $s = 0 \iff b(1) = 0$ using a proof by construction. It follows from Eqs. 6-10 and Eq. 17 that the pure strategy defined by $\bar{\pi}_{1,l}(0) = C$ and $\bar{\pi}_{1,l}(b(1)) = S \iff b(1) > 0$ is a best response for the defender against any attacker strategy when $s = 0 \iff b(1) = 0$. Similarly, given $\bar{\pi}_{1,l}$, we get from Eqs. 6-10 and Eq. 18 that the pure strategy defined by $\bar{\pi}_{2,l}(0, b(1)) = C$ and $\bar{\pi}_{2,l}(1, b(1)) = S$ for all $b(1) \in [0,1]$ is a best response for the attacker. Hence, $(\bar{\pi}_{1,l}, \bar{\pi}_{2,l})$ is a pure Nash equilibrium (see Eq. 19). □

Appendix B
Proof of Theorem 1.B.

Given the POSG Γ introduced in Section III and a fixed attacker strategy $\pi_{2,l}$, the best response strategy of the defender $\tilde{\pi}_{1,l} \in B_1(\pi_{2,l})$ is an optimal strategy in a POMDP $\mathcal{M}^P$ (see Section IV). Hence, it is sufficient to show that there exists an optimal strategy $\pi^*_{1,l}$ in $\mathcal{M}^P$ that satisfies Eq. 26. The conditions for Eq. 26 to hold and the proof are given in our previous work [11][Theorem 1.C]. Since $f_{O|s}$ is TP2 by assumption and all of the remaining conditions hold by definition of Γ, the result follows. □

Appendix C
Proof of Theorem 1.C.

Given the POSG Γ introduced in Section III and a fixed defender strategy $\pi_{1,l}$, the best response strategy of the attacker $\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$ is an optimal strategy in an MDP $\mathcal{M}$ (see Section IV). Hence, it is sufficient to show that there exists an optimal strategy $\pi^*_{2,l}$ in $\mathcal{M}$ that satisfies Eqs. 27-27. To prove this, we use properties of $\mathcal{M}$'s value function $V^*_{\pi_{1,l},l}$.

We use the value iteration algorithm to establish properties of $V^*_{\pi_{1,l},l}$ [38], [41]. Let $V^k_{\pi_{1,l},l}$, $\mathscr{S}^{k,(2)}_{s,l,\pi_{1,l}}$, and $\mathscr{C}^{k,(2)}_{s,l,\pi_{1,l}}$, denote the value function, the stopping set, and the continuation set at iteration k of the value iteration algorithm, respectively. Then, $\lim_{k\to\infty} V^k_{\pi_{1,l},l} = V^*_{\pi_{1,l},l}$, $\lim_{k\to\infty} \mathscr{S}^{k,(2)}_{s,l,\pi_{1,l}} = \mathscr{S}^{(2)}_{s,l,\pi_{1,l}}$, and $\lim_{k\to\infty} \mathscr{C}^{k,(2)}_{s,l,\pi_{1,l}} = \mathscr{C}^{(2)}_{s,l,\pi_{1,l}}$ [38], [41]. We define $V^0_{\pi_{1,l},l}\big((s, b(1))\big) = 0$ for all $b(1) \in [0,1]$, $s \in \mathcal{S}$ and $l \in \{1, \ldots, L\}$.

Towards the proof of Theorem 1.C, we state the following six lemmas.

Lemma 1. *Given any defender strategy $\pi_{1,l}$, $V^*_{\pi_{1,l},2}\big(s, b(1)\big) \geq 0$ for all $s \in \mathcal{S}$ and $b(1) \in [0,1]$.*

Proof. Consider $\bar{\pi}_{2,l}$ defined by $\bar{\pi}_{2,l}(0,\cdot) = C$ and $\bar{\pi}_{2,l}(1,\cdot) = S$. Then it follows from Eqs. 6-10 that for any $\pi_{1,l}$, $s \in \mathcal{S}$ and $b(1) \in [0,1]$, the following holds: $V^{\bar{\pi}_{2,l}}_{\pi_{1,l},l}(s, b(1)) \geq 0$. By optimality, $V^{\bar{\pi}_{2,l}}_{\pi_{1,l},l}(s, b(1)) \leq V^*_{\pi_{1,l},l}(s, b(1))$. Hence, $V^*_{\pi_{1,l},l}(s, b(1)) \geq 0$. □

Lemma 2. *$V^*_{\pi_{1,l},l}\big(1, b(1)\big)$ is non-increasing with $\pi_{1,l}(S|b(1))$ and non-decreasing with $l \in \{1, \ldots, L\}$.*

Proof. We prove this statement by mathematical induction. For $k = 0$, we know from Eqs. 6-10 that $V^0_{\pi_{1,l},l}\big(1,\, b(1)\big)$ is non-increasing with $\pi_{1,l}(S|b(1))$ and non-decreasing with l.

For $k > 0$, $V^k_{\pi_{1,l},l}$ is given by:

$$V^k_{\pi_{1,l},l}\big(1,b(1)\big) = \max\Big[0, -R\big(1,(C,a^{(1)})\big) \qquad (33)$$
$$+ (1-\phi_l)\sum_o f_O(o|1)V^{k-1}_{l-a^{a(1)}}\big(1,b(1)\big)\Big]$$

The first term in the maximization in Eq. 33 is trivially non-increasing with $\pi_{1,l}(S|b(1))$ and non-decreasing with l. Assume by induction that the conditions hold for $V^{k-1}_{\pi_{1,l},l}\big(s, b(1)\big)$. Then the second term in Eq. 33 is non-increasing with $\pi_{1,l}(S|b(1))$ and non-decreasing with l by Eqs. 6-10 and the induction hypothesis. Hence, $V^k_{\pi_{1,l},l}\big(s, b(1)\big)$ is non-increasing with $\pi_{1,l}(S|b(1))$ and non-decreasing with l for all $k \geq 0$. □

Lemma 3. *If f_O is TP2 and $\pi_{1,l}(S|b(1))$ is increasing with $b(1)$, then $V_{\pi_{1,l},l}(b(1), 1) \geq \sum_o f_O(o|1)V_{\pi_{1,l},l}(1, b^o(1))$, where $b^o(1)$ denotes $b(1)$ updated with Eq. 14 after observing $o \in \mathcal{O}$.*

Proof. Since f_O is TP2, it follows from [41, Theorem 10.3.1, pp. 225,238] and [11, Lemma 4, pp. 12] that given two beliefs $b'(1) \geq b(1)$ and two observations $o \geq \bar{o}$, the following holds for any $k \in \mathcal{O}$ and $l_t \in \{1, \dots, L\}$: $b'^{,o}(1) \geq b^o(1)$, $\mathbb{P}[o \geq k|b'(1)] \geq \mathbb{P}[o \geq k|b(1)]$, and $b^o_a(1) \geq b^{\bar{o}}_a(1)$.

Since $\pi_{1,l}$ is increasing with $b(1)$ and $V_{\pi_{1,l},l}(b(1), 1)$ is decreasing with $b(1)$ (Lemma 2), it follows that $\mathbb{E}_o[b^o(1)] \geq b(1)$, and thus $V_{\pi_{1,l},l}(b(1), 1) \geq \sum_o f_O(o|1)V_{\pi_{1,l},l}(1, b^o(1))$. □

Lemma 4. *If f_O is TP2, $\pi_{1,l}(S|b(1)) = 1$, and $\pi_{1,l}(S|b(1))$ is increasing with $b(1)$, then $V^*_{\pi_{1,l},2}\big(s,b(1)\big)= 0$ and for any $\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$, $\tilde{\pi}_{2,l}(1,b(1))= 1$.*

Proof. From Eqs. 22-25 we know that $\tilde{\pi}_{2,l}(1, b(1)) = 1$ iff:

$$R_{st}/l + (\phi_l - 1)\sum_o f_O(o|1)V^*_{\pi_{1,l},l-a^{(1)}}(1, b^o(1)) \geq 0 \quad (34)$$

$R_{st} \geq 0$ (see Section III). Since f_O is TP2, $\pi_{1,l}(S|b(1)) = 1$, and $\pi_{1,l}(S|b(1))$ is increasing with $b(1)$, we have by Lemma 3 that $\mathbb{E}_o[\pi_{1,l}(S|b^o(1)) = 1]$. The second term in the left-hand side of Eq. 34 is thus zero. Hence, the inequality holds and $\tilde{\pi}_{2,l}(1, b(1)) = 1$, which implies that $V^*_{\pi_{1,l},2}\big(s, b(1)\big) = 0$. □

Lemma 5. *Given any defender strategy $\pi_{1,l}$, if $\pi^*_{2,l}(1, b(1)) = S$, then $\pi^*_{2,l}(0, b(1)) = C$.*

Proof. $\pi^*_{2,l}(1,b(1))= S$ implies that $V^*_{\pi_{1,l},l}(1,b(1))= 0$. Hence, by Lemma 3 we get that:

$$(1-\phi_l)\sum_{o\in\mathcal{O}} f_O(o|1)V^*_{\pi_{1,l},l}(1, b^o) \leq 0 \qquad (35)$$
$$\implies \sum_{o\in\mathcal{O}} f_O(o|1)V^*_{\pi_{1,l},l}(1, b^o) \leq \sum_{o\in\mathcal{O}} f_O(o|0)V^*_{\pi_{1,l},l}(0, b^o)$$
$$\implies \pi^*_{2,l}(0, b(1)) = C$$

□

Lemma 6. *If $\pi_{1,l}(S|b(1))$ is non-decreasing with $b(1)$ and f_O is TP2, then $V^*_{\pi_{1,l},l}\big(0, b(1)\big) - V^*_{\pi_{1,l},l}\big(1, b(1)\big)$ is non-decreasing with $\pi_{1,l}(S|b(1))$.*

Proof. We prove this statement by mathematical induction. Let $W^k_{\pi_{1,l},l}(b(1)) = V^k_{\pi_{1,l},l}\big(0, b(1)\big) - V^k_{\pi_{1,l},l}\big(1, b(1)\big)$. For $k = 0$, it follows from Eqs. 6-10 that $W^0_{\pi_{1,l},l}(b(1))$ is non-decreasing with $\pi_{1,l}(S|b(1)) \in [0, 1]$. Assume by induction that the conditions hold for $W^{k-1}_{\pi_{1,l},l}(b(1))$. We show that then the conditions hold also for $W^k_{\pi_{1,l},l}(b(1))$.

There are three cases to consider:

- If $b(1) \in \mathscr{S}^{k,(2)}_{0,l,\pi_{1,l}} \cap \mathscr{C}^{k,(2)}_{1,l,\pi_{1,l}}$, then:

$$W^k_{\pi_{1,l},l}(b(1)) = R_{int}+ \qquad (36)$$
$$\pi_{1,l}(S|b(1))(R_{st}/l - R_{cost}/l - R_{int})$$

  which is non-decreasing with $\pi_{1,l}(S|b(1))$ since $R_{st}/l - R_{cost}/l - R_{int} \geq 0$ (see Section III).

- If $b(1) \in \mathscr{C}^{k,(2)}_{0,l,\pi_{1,l}} \cap \mathscr{C}^{k,(2)}_{1,l,\pi_{1,l}}$, then:

$$W^k_{\pi_{1,l},l}(b(1)) = \pi_{1,l}(S|b(1))\Big(R_{st}/l - R_{cost}/l \qquad (37)$$
$$- R_{int}\Big) + V^{k-1}_{\pi_{1,l},l}\big(1, b(1)\big)\Big) + R_{int} + \sum_o f_O(o|0)$$
$$V^k_{\pi_{1,l},l}\big(0, b^o(1)\big) - (1-\phi_l)f_O(o|1)V^k_{\pi_{1,l},l}\big(1, b^o(1)\big)$$

  The first term is non-decreasing with $\pi_{1,l}(S|b(1))$ since $R_{st}/l - R_{cost}/l - R_{int} \geq 0$ (see Section III) and $V^{k-1}_{\pi_{1,l},l}\big(1, b(1)\big) \geq 0$ (it is a consequence of Lemma 1 and Eqs. 22-25). The second term is non-decreasing with $\pi_{1,l}(S|b(1))$ by the induction hypothesis and the assumption that f_O is TP2.

- If $b(1) \in \mathscr{C}^{k,(2)}_{0,l,\pi_{1,l}} \cap \mathscr{S}^{k,(2)}_{1,l,\pi_{1,l}}$, then:

$$W^k_{\pi_{1,l},l}(b(1)) = \pi_{1,l}(S|b(1))(-R_{cost}/l) \qquad (38)$$
$$+ \sum_o f_O(o|0)V^k_{\pi_{1,l},l}\big(0, b^o(1)\big)$$
$$= \pi_{1,l}(S|b(1))(-R_{cost}/l) + \sum_o f_O(o|0)\cdot$$
$$V^k_{\pi_{1,l},l}\big(0, b^o(1)\big) - (1-\phi_l)f_O(o|1)V^k_{\pi_{1,l},l}\big(1, b^o(1)\big)$$

  The first term is non-decreasing with $\pi_{1,l}(S|b(1))$ since $-R_{cost}/l \geq 0$. The second term is non-decreasing with $\pi_{1,l}(S|b(1))$ by the induction hypothesis and the assumption that f_O is TP2. The second equality in Eq. 38 follows from Lemma 3 and because $b(1) \in \mathscr{S}^{k,(2)}_{1,l,\pi_{1,l}}$.

The other cases, e.g. $b(1) \in \mathscr{S}^{k,(2)}_{0,l,\pi_{1,l}} \cap \mathscr{S}^{k,(2)}_{1,l,\pi_{1,l}}$, can be discarded due to Lemma 5. Hence, $W^k_{\pi_{1,l},l}(b(1))$ is non-decreasing with $\pi_{1,l}(S|b(1))$ for all $k \geq 0$. □

We now use Lemmas 1-6 to prove Theorem 1.C. The main idea behind the proof is to show that the stopping sets in state $s = 1$ have the form: $\mathscr{S}^2_{1,l,\pi_{1,l}} = [\tilde{\beta}_{1,l}, 1]$, and that the continuation sets in state $s = 0$ have the form: $\mathscr{C}^2_{0,l,\pi_{1,l}} = [\tilde{\beta}_{0,l}, 1]$, for some values $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \dots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0, 1]$.

Proof of Theorem 1.C. We first show that $1 \in \mathscr{S}^{(2)}_{1,l,\pi_{1,l}}$ and that $1 \in \mathscr{C}^{(2)}_{0,l,\pi_{1,l}}$. Since $\pi_{1,l}(S|1) = 1$, by Lemma 4 we have that $1 \in \mathscr{S}^{(2)}_{1,l,\pi_{1,l}}$ and it follows from Eqs. 22-25 that $\tilde{\pi}_{2,l}(0, b(1)) = C$ iff:

$$\sum_o f_O(o|0)V^*_{\pi_{1,l},l-1}(0, b^o(1)) -$$
$$f_O(o|1)V^*_{\pi_{1,l},l-1}(1, b^o(1)) \geq 0 \qquad (39)$$

Game Parameters	**Values**
$R_{st}, R_{cost}, R_{int}$,$\gamma$	20, -2, -1, 0.99
T-FP Parameters	**Values**
$c, \epsilon, \lambda, A, a, N$	10, 0.101, 0.602, 100, 1, 50
NFSP Parameters	**Values**
lr RL, lr SL, batch, # layers	10^{-2},$5 \cdot 10^{-3}$, 64, 2
# neurons, $\mathcal{M}_{RL}$, $\mathcal{M}_{SL}$	128, 2×10^5, 2×10^6,
ϵ, ϵ-decay, η	0.06, 0.001, 0.1
HSVI Parameter	**Value**
ϵ	3

TABLE 4: Hyperparameters of the POSG and the algorithms used for evaluation.

ID (s)	*OS:Services:Exploitable Vulnerabilities*
N_1	Ubuntu20:Snort(community ruleset v2.9.17.1),SSH:-
N_2	Ubuntu20:SSH,HTTP Erl-Pengine,DNS:SSH-pw
N_4	Ubuntu20:HTTP Flask,Telnet,SSH:Telnet-pw
N_{10}	Ubuntu20:FTP,MongoDB,SMTP,Tomcat,TS3,SSH:FTP-pw
N_{12}	Jessie:TS3,Tomcat,SSH:CVE-2010-0426,SSH-pw
N_{17}	Wheezy:Apache2,SNMP,SSH:CVE-2014-6271
N_{18}	Deb9.2:IRC,Apache2,SSH:SQL Injection
N_{22}	Jessie:PROFTPD,SSH,Apache2,SNMP:CVE-2015-3306
N_{23}	Jessie:Apache2,SMTP,SSH:CVE-2016-10033
N_{24}	Jessie:SSH:CVE-2015-5602,SSH-pw
N_{25}	Jessie: Elasticsearch,Apache2,SSH,SNMP:CVE-2015-1427
N_{27}	Jessie:Samba,NTP,SSH:CVE-2017-7494
N_3,N_{11},N_5-N_9	Ubuntu20:SSH,SNMP,PostgreSQL,NTP:-
N_{13-16},N_{19-21},N_{26},N_{28-31}	Ubuntu20:NTP, IRC, SNMP, SSH, PostgreSQL:-

TABLE 5: Configuration of the target infrastructure (Fig. 1).

The left-hand side of the above equation is positive due to the assumption that f_O is TP2 and since $\sum_o f_O(o|0)V^*_{\pi_{1,l},l-1}(0, b^o(1)) \geq 0$ by Lemma 1 and $f_O(o|1)V^*_{\pi_{1,l},l-1}(1,b^o(1)) = 0$ by Lemma 3. Hence, $1 \in \mathscr{C}^{(2)}_{0,l,\pi_{1,l}}$.

Now we show that $\mathscr{S}^2_{1,l,\pi_{1,l}} = [\tilde{\beta}_{1,l}, 1]$ and that $\mathscr{C}^2_{0,l,\pi_{1,l}} = [\tilde{\beta}_{0,l}, 1]$ for some values $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0, 1]$. From Eqs. 22-25 we know that $\tilde{\pi}_{2,l}(1, b(1)) = S$ iff:

$$\begin{aligned} &\mathbb{E}_{\pi_{1,l}}\Big[\mathcal{R}_l\big(1,(a^{(1)},C)\big) \\ &(\phi_l - 1)\sum_o f_O(o|1)V^*_{\pi_{1,l},l-a^{(1)}}(1, b^o(1))\Big] \geq 0 \end{aligned} \tag{40}$$

The first term in the above expectation is increasing with $b(1)$ (Eqs. 6-10). The second term is decreasing with $b(1)$ (Lemma 2). Hence, we conclude that if $\tilde{\pi}_{2,l}(1, b(1)) = S$, then for any $b'(1) \geq b(1)$, $\tilde{\pi}_{2,l}(1, b'(1)) = S$. As a consequence, there exists values $\tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{1,L}$ such that $\mathscr{S}^2_{1,l,\pi_{1,l}} = [\tilde{\beta}_{1,l}, 1]$.

Similarly, from Eqs. 22-25 we know that $\tilde{\pi}_{2,l}(0, b(1)) = C$ iff:

$$\begin{aligned} &\mathbb{E}_{\pi_{1,l}}\Big[\sum_o f_O(o|0)V^*_{\pi_{1,l},l-a^{(1)}}(0, b^o(1)) \\ &\quad - f_O(o|1)V^*_{\pi_{1,l},l-a^{(1)}}(1, b^o(1))\Big] \geq 0 \end{aligned} \tag{41}$$

Since f_O is TP2 and $\pi_{1,l}(S|b(1))$ is increasing with $b(1)$, the left-hand side in the above inequality is decreasing (Lemma 2 and Lemma 6). Hence, we conclude that if $\tilde{\pi}_{2,l}(0, b(1)) = C$, then for any $b'(1) \geq b(1)$, $\tilde{\pi}_{2,l}(0, b'(1)) = C$. As a result, there exists values $\tilde{\beta}_{0,1}, \ldots, \tilde{\beta}_{0,L}$ such that $\mathscr{C}^2_{0,l,\pi_{1,l}} = [\tilde{\beta}_{0,l}, 1]$. □

Appendix D
Hyperparameters

The hyperparameters used for the evaluation are listed in Table 4 and were obtained through grid search.

Appendix E
Configuration of the Infrastructure in Fig. 1

The configuration of the target infrastructure (Fig. 1) is available in Table 5.

References

[1] A. Fuchsberger, "Intrusion detection systems and intrusion prevention systems," *Inf. Secur. Tech. Rep.*, vol. 10, no. 3, p. 134–139, Jan. 2005.

[2] K.-K. R. Choo, "The cyber threat landscape: Challenges and future research directions," *Comput. Secur.*, vol. 30, no. 8, p. 719–731, 2011.

[3] P. Johnson, R. Lagerström, and M. Ekstedt, "A meta language for threat modeling and attack simulations," in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ser. ARES 2018, New York, NY, USA, 2018.

[4] M. Rasouli, E. Miehling, and D. Teneketzis, "A supervisory control approach to dynamic cyber-security," in *Decision and Game Theory for Security*. Cham: Springer International Publishing, 2014, pp. 99–117.

[5] E. Miehling, M. Rasouli, and D. Teneketzis, *Control-Theoretic Approaches to Cyber-Security*. Cham: Springer International Publishing, 2019, pp. 12–28.

[6] R. Bronfman-Nadas, N. Zincir-Heywood, and J. T. Jacobs, "An artificial arms race: Could it improve mobile malware detectors?" in *2018 Network Traffic Measurement and Analysis Conference (TMA)*, 2018.

[7] N. Wagner, C. c. Şahin, M. Winterrose, J. Riordan, J. Pena, D. Hanson, and W. W. Streilein, "Towards automated cyber decision support: A case study on network segmentation for security," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, 2016, pp. 1–10.

[8] C. Wagner, A. Dulaunoy, G. Wagener, and A. Iklody, "Misp: The design and implementation of a collaborative threat intelligence sharing platform," in *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, ser. WISCS '16. New York, NY, USA: Association for Computing Machinery, 2016, p. 49–56.

[9] T. Alpcan and T. Basar, *Network Security: A Decision and Game-Theoretic Approach*, 1st ed. USA: Cambridge University Press, 2010.

[10] S. Sarıtaş, E. Shereen, H. Sandberg, and G. Dán, "Adversarial attacks on continuous authentication security: A dynamic game approach," in *Decision and Game Theory for Security*, Cham, 2019, pp. 439–458.

[11] K. Hammar and R. Stadler, "Intrusion prevention through optimal stopping," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022, https://ieeexplore.ieee.org/document/9779345.

[12] ——, "Finding effective security strategies through reinforcement learning and Self-Play," in *International Conference on Network and Service Management (CNSM 2020)*, Izmir, Turkey, 2020.

[13] ——, "Learning intrusion prevention policies through optimal stopping," in *International Conference on Network and Service Management (CNSM 2021)*, Izmir, Turkey, 2021, https://arxiv.org/pdf/2106.07160.pdf.

[14] A. Wald, *Sequential Analysis*. Wiley and Sons, New York, 1947.

[15] E. Dynkin, "A game-theoretic version of an optimal stopping problem," *Dokl. Akad. Nauk SSSR*, vol. 385, pp. 16–19, 1969.

[16] A. N. Shiryaev, "On optimum methods in quickest detection problems," *Theory Probab. Appl.*, vol. 8, no. 1, pp. 22–46, 1963.

[17] K. Hammar and R. Stadler, "A software framework for building self-learning security systems," 2022, https://www.youtube.com/watch?v=18P7MjPKNDg.

[18] F. M. Zennaro and L. Erdodi, "Modeling penetration testing with reinforcement learning using capture-the-flag challenges and tabular q-learning," *CoRR*, 2020, https://arxiv.org/abs/2005.12632.

[19] A. Ridley, "Machine learning for autonomous cyber defense," 2018, the Next Wave, Vol 22, No.1 2018.

[20] K. Tran, A. Akella, M. Standen, J. Kim, D. Bowman, T. Richer, and C.-T. Lin, "Deep hierarchical reinforcement agents for automated penetration testing," 2021, https://arxiv.org/abs/2109.06449.

[21] R. Gangupantulu, T. Cody, P. Park, A. Rahman, L. Eisenbeiser, D. Radke, and R. Clark, "Using cyber terrain in reinforcement learning for penetration testing," 2021, https://arxiv.org/abs/2108.07124.

[22] T. V. Phan and T. Bauschert, "Deepair: Deep reinforcement learning for adaptive intrusion response in software-defined networks," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[23] Z. Hu, M. Zhu, and P. Liu, "Adaptive cyber defense against multi-stage attacks using learning-based pomdp," *ACM Trans. Priv. Secur.*, vol. 24, no. 1, Nov. 2020.

[24] M. N. Kurt, O. Ogundijo, C. Li, and X. Wang, "Online cyber-attack detection in smart grid: A reinforcement learning approach," *IEEE Transactions on Smart Grid*, vol. 10, no. 5, pp. 5174–5185, 2019.

[25] R. Elderman, L. J. J. Pater, A. S. Thie, M. M. Drugan, and M. Wiering, "Adversarial reinforcement learning in a cyber security simulation," in *ICAART*, 2017.

[26] J. Schwartz, H. Kurniawati, and E. El-Mahassni, "Pomdp + information-decay: Incorporating defender's behaviour in autonomous penetration testing," *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 30, no. 1, pp. 235–243, Jun. 2020.

[27] W. Blum, "Gamifying machine learning for stronger security and ai models," 2021, https://www.microsoft.com/security/blog/2021/04/08/gamifying-machine-learning-for-stronger-security-and-ai-models/.

[28] M. Zhu, Z. Hu, and P. Liu, "Reinforcement learning algorithms for adaptive cyber defense against heartbleed," in *Proceedings of the First ACM Workshop on Moving Target Defense*, ser. MTD '14. New York, NY, USA: Association for Computing Machinery, 2014, p. 51–58.

[29] M. van Dijk, A. Juels, A. Oprea, and R. L. Rivest, "Flipit: The game of "stealthy takeover"," *Journal of Cryptology*, no. 4, Oct 2013.

[30] L. Huang and Q. Zhu, "A dynamic games approach to proactive defense strategies against advanced persistent threats in cyber-physical systems," *Computers & Security*, vol. 89, p. 101660, 11 2019.

[31] S. Sengupta, A. Chowdhary, D. Huang, and S. Kambhampati, *General Sum Markov Games for Strategic Detection of Advanced Persistent Threats Using Moving Target Defense in Cloud Networks*, 10 2019.

[32] J. F. Nash, "Non-cooperative games," *Annals of Mathematics*, vol. 54, pp. 286–295, 1951.

[33] J. von Neumann, "Zur Theorie der Gesellschaftsspiele. (German) [On the theory of games of strategy]," vol. 100, pp. 295–320, 1928.

[34] L. S. Shapley, "Stochastic games," *Proceedings of the National Academy of Sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.

[35] J. Hespanha and M. Prandini, "Nash equilibria in partial-information games on markov chains," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, vol. 3, 2001.

[36] R. Bellman, "A markovian decision process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957.

[37] K. Horák, B. Bosanský, V. Kovarík, and C. Kiekintveld, "Solving zero-sum one-sided partially observable stochastic games," *CoRR*, vol. abs/2010.11243, 2020.

[38] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed. USA: John Wiley and Sons, Inc., 1994.

[39] G. Peskir and A. Shiryaev, *Optimal stopping and free-boundary problems*, ser. Lectures in mathematics (ETH Zürich). Springer, 2006.

[40] Y. Chow, H. Robbins, and D. Siegmund, "Great expectations: The theory of optimal stopping," 1971.

[41] V. Krishnamurthy, *Partially Observed Markov Decision Processes: From Filtering to Controlled Sensing*. Cambridge University Press, 2016.

[42] T. Nakai, "The problem of optimal stopping in a partially observable markov chain," *Journal of Optimization Theory and Applications*, vol. 45, no. 3, pp. 425–442, Mar 1985.

[43] V. Krishnamurthy, A. Aprem, and S. Bhatt, "Multiple stopping time pomdps: Structural results & application in interactive advertising on social media," *Automatica*, vol. 95, pp. 385–398, 2018.

[44] G. W. Brown, "Iterative solution of games by fictitious play," 1951, activity analysis of production and allocation.

[45] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, UK: Cambridge University Press, 2009.

[46] J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," *IEEE TRANSACTIONS ON AUTOMATIC CONTROL*, vol. 37, no. 3, pp. 332–341, 1992.

[47] J. Spall, "Implementation of the simultaneous perturbation algorithm for stochastic optimization," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 34, no. 3, pp. 817–823, 1998.

[48] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux journal*, vol. 2014, no. 239, p. 2, 2014.

[49] S. Hemminger, "Network emulation with netem," *Linux Conf*, 2005.

[50] T. Kushida and Y. Shibata, "Empirical study of inter-arrival packet times and packet losses," in *Proceedings of the 22nd International Conference on Distributed Computing Systems*, 2002, p. 233–240.

[51] V. Paxson, "End-to-end internet packet dynamics," in *IEEE/ACM Transactions on Networking*, 1997, pp. 277–292.

[52] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *The Bell System Technical Journal*, vol. 42, no. 5, pp. 1977–1997, 1963.

[53] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society: Series B*, vol. 39, pp. 1–38, 1977.

[54] J. Kreps, "Kafka : a distributed messaging system for log processing," 2011.

[55] Google, "Google remote procedure call," 2022.

[56] K. Hammar and R. Stadler, "gym-optimal-intrusion-response," 2021, https://github.com/Limmen/gym-optimal-intrusion-response.

[57] F. Timbers, E. Lockhart, M. Schmid, M. Lanctot, and M. Bowling, "Approximate exploitability: Learning a best response in large games," *CoRR*, vol. abs/2004.09677, 2020.

[58] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *CoRR*, vol. abs/1603.01121, 2016.

[59] K. Horák, B. Bošanský, and M. Pěchouček, "Heuristic search value iteration for one-sided partially observable stochastic games," *Proceedings of the AAAI Conference on Artificial Intelligence*, Feb. 2017.

[60] K. Horák, "Scalable algorithms for solving stochastic games with limited partial observability," Ph.D. dissertation, 2019.

[61] M. Roesch, "Snort - lightweight intrusion detection for networks," in *Proceedings of the 13th USENIX Conference on System Administration*, ser. LISA '99. USA: USENIX Association, 1999, p. 229–238.

[62] Khraisat *et al.*, "Survey of intrusion detection systems: techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, p. 20, 2019.

[63] J. Dromard, G. Roudière, and P. Owezarski, "Online and scalable unsupervised network anomaly detection method," *IEEE Transactions on Network and Service Management*, vol. 14, no. 1, pp. 34–47, 2017.

[64] A. G. Tartakovsky, B. L. Rozovskii, R. B. Blažek, and H. Kim, "Detection of intrusions in information systems by sequential change-point methods," *Statistical Methodology*, vol. 3, no. 3, 2006.

[65] C. J. Fung, J. Zhang, and R. Boutaba, "Effective acquaintance management based on bayesian learning for distributed intrusion detection networks," *IEEE Transactions on Network and Service Management*, vol. 9, no. 3, pp. 320–332, 2012.

[66] C. J. Fung, J. Zhang, I. Aib, and R. Boutaba, "Dirichlet-based trust management for effective collaborative intrusion detection networks," *IEEE Transactions on Network and Service Management*, 2011.

[67] S. Huang *et al.*, "Hitanomaly: Hierarchical transformers for anomaly detection in system log," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2064–2076, 2020.

[68] M. Tambe, *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*, 1st ed. USA: Cambridge University Press, 2011.

[69] C. J. Fung and R. Boutaba, *Intrusion Detection Networks - A Key to Collaborative Security*. CRC Press, 2013.

[70] L. Buttyan and J.-P. Hubaux, *Security and Cooperation in Wireless Networks: Thwarting Malicious and Selfish Behavior in the Age of Ubiquitous Computing*. USA: Cambridge University Press, 2007.

[71] N. Dhir, H. Hoeltgebaum, N. Adams, M. Briers, A. Burke, and P. Jones, "Prospective artificial intelligence approaches for active cyber defence," *CoRR*, vol. abs/2104.09981, 2021, https://arxiv.org/abs/2104.09981.

[72] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *CoRR*, 2019, http://arxiv.org/abs/1906.05799.

[73] Y. Huang, L. Huang, and Q. Zhu, "Reinforcement learning for feedback-enabled cyber resilience," *Annual Reviews in Control*, 2022.

[74] E. Miehling, M. Rasouli, and D. Teneketzis, "A pomdp approach to the dynamic defense of large-scale cyber networks," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 10, 2018.

[75] J. Gabirondo-López, J. Egaña, J. Miguel-Alonso, and R. Orduna Urrutia, "Towards autonomous defense of sdn networks using muzero based intelligent agents," *IEEE Access*, vol. 9, pp. 107 184–107 199, 2021.

[76] I. Akbari, E. Tahoun, M. A. Salahuddin, N. Limam, and R. Boutaba, "Atmos: Autonomous threat mitigation in sdn using reinforcement learning," in *NOMS IEEE/IFIP Network Operations and Management Symposium*, 2020, pp. 1–9.

[77] Y. Liu *et al.*, "Deep reinforcement learning based smart mitigation of ddos flooding in software-defined networks," in *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2018, pp. 1–6.

[78] K. Hammar and R. Stadler, "A system for interactive examination of learned security policies," 2022, https://arxiv.org/abs/2204.01126.

[79] A. Aydeger, M. H. Manshaei, M. A. Rahman, and K. Akkaya, "Strategic defense against stealthy link flooding attacks: A signaling game approach," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 1, pp. 751–764, 2021.

[80] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, ser. Hotnets-IX. New York, NY, USA: Association for Computing Machinery, 2010.

[81] M. Standen, M. Lucas, D. Bowman, T. J. Richer, J. Kim, and D. Marriott, "Cyborg: A gym for the development of autonomous cyber agents," *CoRR*, https://arxiv.org/abs/2108.09118.

[82] A. Molina-Markham, C. Miniter, B. Powell, and A. Ridley, "Network environment design for autonomous cyberdefense," 2021, https://arxiv.org/abs/2103.07583.

[83] K. Durkota, V. Lisy, B. Bošansky, and C. Kiekintveld, "Optimal network security hardening using attack graph games," in *Proceedings of the 24th International Conference on Artificial Intelligence*, 2015.

[84] K. Horák, B. Bosanský, P. Tomásek, C. Kiekintveld, and C. A. Kamhoua, "Optimizing honeypot strategies against dynamic lateral movement using partially observable stochastic games," *Comput. Secur.*, vol. 87, 2019.

[85] R. Píbil, V. Lisý, C. Kiekintveld, B. Bošanský, and M. Pěchouček, "Game theoretic model of strategic honeypot selection in computer networks," in *Decision and Game Theory for Security*, J. Grossklags and J. Walrand, Eds., 2012.

[86] O. Vaněk, Z. Yin, M. Jain, B. Bošanský, M. Tambe, and M. Pěchouček, "Game-theoretic resource allocation for malicious packet detection in computer networks," in *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, 2012.

[87] O. Tsemogne, Y. Hayel, C. Kamhoua, and G. Deugoue, *Partially Observable Stochastic Games for Cyber Deception Against Network Epidemic*, 12 2020, pp. 312–325.

[88] K. C. Nguyen, T. Alpcan, and T. Basar, "Stochastic games for security in networks with interdependent nodes," in *2009 International Conference on Game Theory for Networks*, 2009, pp. 697–703.

[89] A. Laszka, W. Abbas, S. S. Sastry, Y. Vorobeychik, and X. Koutsoukos, "Optimal thresholds for intrusion detection systems," in *Proceedings of the Symposium and Bootcamp on the Science of Security*, 2016.

[90] T. Alpcan and T. Basar, "A game theoretic analysis of intrusion detection in access control systems," in *2004 43rd IEEE Conference on Decision and Control (CDC) (IEEE Cat. No.04CH37601)*, 2004.

[91] Q. Zhu and T. Başar, "Dynamic policy-based ids configuration," in *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, 2009.

[92] M. Alario-Nazaret, J. P. Lepeltier, and B. Marchal, "Dynkin games," in *Stochastic Differential Systems*, M. Kohlmann and N. Christopeit, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1982, pp. 23–32.

[93] E. Solan and N. Vieille, "Deterministic multi-player dynkin games," 2002.

[94] J. Lempa and P. Matomäki, "A dynking game with asymmetric information," 2010.

[95] E. Ekström, K. Glover, and M. Leniec, "Dynkin games with heterogeneous beliefs," *Journal of Applied Probability*, no. 1, 2017.

[96] T. Noe, "Capital structure and signaling game equilibria," *Review of Financial Studies*, vol. 1, no. 4, pp. 331–355, 1988.

[97] M. H. Manshaei, Q. Zhu, T. Alpcan, T. Basar, and J.-P. Hubaux, "Game theory meets network security and privacy," *ACM Comput. Surv.*, vol. 45, no. 3, pp. 25:1–25:39, Jul. 2013.