Learning Near-Optimal Intrusion Responses Against Dynamic Attackers

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
January 18, 2023

***Abstract*—We study automated intrusion response and formulate the interaction between an attacker and a defender as an optimal stopping game where attack and defense strategies evolve through reinforcement learning and self-play. The game-theoretic modeling enables us to find defender strategies that are effective against a dynamic attacker, i.e. an attacker that adapts its strategy in response to the defender strategy. Further, the optimal stopping formulation allows us to prove that optimal strategies have threshold properties. To obtain near-optimal defender strategies, we develop Threshold Fictitious Self-Play (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. The experimental part of this investigation includes two systems: a simulation system where defender strategies are incrementally learned and an emulation system where statistics are collected that drive simulation runs and where learned strategies are evaluated. We argue that this approach can produce effective defender strategies for a practical IT infrastructure.**

***Index Terms*—Cybersecurity, network security, automated security, intrusion response, optimal stopping, Dynkin games, reinforcement learning, game theory, Markov decision process, MDP, POMDP.**

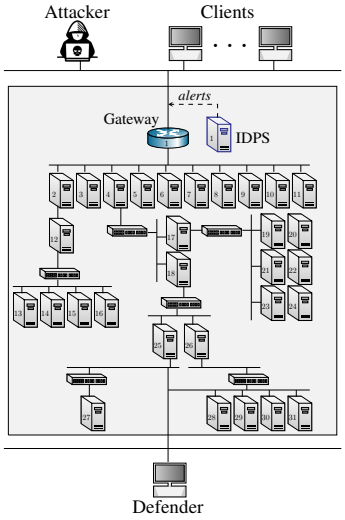

Fig. 1: The IT infrastructure and the actors in the intrusion response use case.

I. Introduction

An organization's security strategy has traditionally been defined, implemented, and updated by domain experts [1]. This approach can provide basic security for an organization's communication and computing infrastructure. As infrastructure update cycles become shorter and attacks increase in sophistication, meeting the security requirements becomes increasingly difficult. To address this challenge, significant efforts have started to automate the process of obtaining security strategies [2]. Examples of this research include: computation of defender strategies using dynamic programming and control theory [3], [4]; computation of exploits and corresponding defenses through evolutionary methods [5], [6]; computation of defender strategies through game-theoretic methods [7], [8]; use of machine learning techniques to estimate model parameters and strategies [9]–[11]; automated creation of threat models [12]; and identification of infrastructure vulnerabilities through attack simulations and threat intelligence [13], [14].

A promising new direction of research is automatically learning security strategies through reinforcement learning methods, whereby the problem of finding security strategies is modeled as a Markov decision problem and strategies are learned through simulation (see surveys [15], [16]). While encouraging results have been obtained following this approach [9]–[11], [17]–[54], key challenges remain [55]. Chief among them is narrowing the gap between the environment where strategies are evaluated and a scenario playing out in a real system. Most of the results obtained so far are limited to simulation environments, and it is not clear how they generalize to practical IT infrastructures. Another challenge is to obtain security strategies that are effective against a dynamic attacker, i.e. an attacker that adapts its strategy in response to the defender strategy. Most of the prior work have used reinforcement learning to find effective defender strategies against static attackers, and little is known about the found strategies' performance against a dynamic attacker.

In this paper, we address the above challenges and present a novel framework to automatically learn a defender strategy against a dynamic attacker. We apply this framework to an *intrusion response* 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 an attacker while providing services to a client population.

We formulate the intrusion response use case as an *opti-*

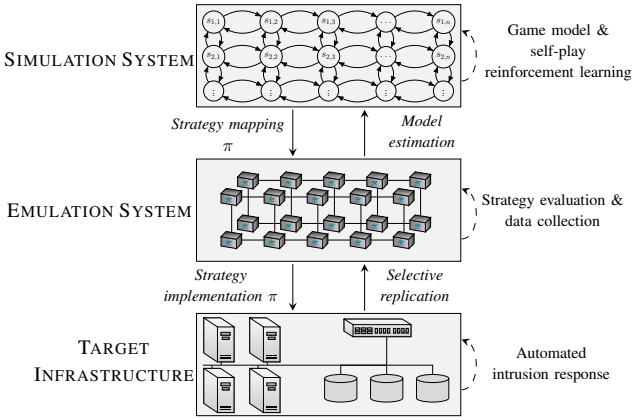

Fig. 2: Our framework for finding and evaluating intrusion response strategies [9].

mal stopping game, namely a stochastic game where both players face an optimal stopping problem [56]–[58]. This formulation enables us to gain insight into the structure of optimal strategies, which we prove to have threshold properties. To obtain effective defender strategies, we use reinforcement learning and self-play. Based on the threshold properties, we design Threshold Fictitious Self-Play (T-FP), an efficient algorithm that iteratively computes near-optimal defender strategies against a dynamic attacker.

Our method for learning and evaluating strategies for a given infrastructure 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. Such runs produce system measurements and logs, from which we estimate infrastructure statistics, which then are used to instantiate the simulation model.

Second, we build a *simulation system* where game episodes are simulated and strategies are incrementally learned through self-play. Learned strategies are extracted from the simulation system and evaluated in the emulation system.

Two benefits of this method are: (*i*) that the emulation system allows evaluating strategies without affecting operational workflows on the target infrastructure; and (*ii*) that the simulation system enables efficient and rapid learning of strategies. (A video demonstration of the software framework that implements the emulation and simulation systems is available at [59].)

We make three contributions with this paper. First, we formulate intrusion response as an optimal stopping game between an attacker and a defender. This novel formulation allows us a) to derive and prove structural properties of optimal strategies; and b) to find defender strategies that are effective against an attacker with a dynamic strategy. We thus address a key limitation of many related works, which only consider static attackers [9], [11], [17], [20], [21], [23], [25]–[27], [31], [36], [37], [39], [40], [42], [48], [51]–[53], [60]–[65]. Second, we propose T-FP, an efficient reinforcement learning algorithm that exploits threshold 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 drawback in related research that relies solely on simulations to learn and evaluate strategies [8], [10], [11], [18]–[27], [33]–[35], [38], [39], [43]–[46], [50], [52], [54], [60], [62]–[72].

We believe that this paper provides a foundation for the next generation of security systems, including Intrusion Prevention Systems (IPSs) (e.g. Trellix [73]), Intrusion Response Systems (IRSs) (e.g. Wazuh [74]), and Intrusion Detection Systems (IDSs) (e.g. Snort [75]). The optimal stopping strategies computed through our framework can be used in these systems to decide at which point in time an automated response action should be triggered or at which point in time a human operator should be alerted to take action.

The work in this paper builds on our earlier results in automated intrusion response [9], [11], [76]. Specifically, this paper can be seen as a generalization of the work in [9], where we investigate intrusion response against a static attacker. As explained in this paper, intrusion response against a dynamic attacker requires a fundamentally different and more complex framework, as well as new algorithms to compute defender strategies. An extended abstract of this paper was presented at the "Machine learning for cyber security" workshop at the International Conference on Machine Learning (ICML) 2022 [76]. The workshop does not have official proceedings.

II. The Intrusion Response Use Case

We consider an intrusion response 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 providing services to a client population (Fig. 1). The infrastructure includes a set of servers that run the services and an Intrusion Detection and Prevention System (IDPS) that logs events in real-time. Clients access the services through a public gateway, which is also open to the attacker.

The attacker's goal is to intrude on the infrastructure and compromise its servers. To achieve this, the attacker explores the infrastructure through reconnaissance and exploits vulnerabilities while avoiding detection by the defender. The attacker decides when to start an intrusion and may stop the intrusion at any moment. During the intrusion, the attacker follows a pre-defined strategy. When deciding the time to start or stop an intrusion, the attacker considers both the gain of compromising additional servers and the risk of detection. The optimal strategy for the attacker is to compromise as many servers as possible without being detected.

The defender continuously monitors the infrastructure through accessing and analyzing IDPS alerts and other statistics. It can take a fixed number of defensive actions, each of which has a cost and a chance of stopping an ongoing attack. An example of a defensive action is to drop network traffic that triggers IDPS alerts of a certain priority. The defender takes defensive actions in a pre-determined order, starting with the action that has the lowest cost. The final action blocks all external access to the gateway, which disrupts any intrusion as well as the services to the clients.

When deciding the time for taking a defensive action, the defender balances two objectives: (*i*) maintain services to its clients; and (*ii*), stop a possible intrusion at the lowest cost. The optimal strategy for the defender is to monitor the infrastructure and maintain services until the moment when the attacker enters through the gateway, at which time the attack must be stopped at minimal cost through defensive actions. The challenge for the defender is to identify this precise moment.

III. Formalizing The Intrusion Response Use Case

We formulate the above intrusion response use case as a partially observed stochastic game. The attacker wins the game when it can intrude on the infrastructure and hide its actions from the defender. Similarly, the defender wins the game when it manages to stop an intrusion. It is 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 Detection and Prevention System (IDPS) 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. This means that the defender has to find strategies that are effective against an opponent that has more knowledge than itself.

The reward function of the game encodes the defender's objective. An optimal defender strategy *maximizes* the reward when facing an attacker with an optimal strategy, i.e. a worst-case attacker. Similarly, an optimal attacker strategy *minimizes* the reward when facing a worst-case defender. Such a pair of optimal strategies is known as a Nash equilibrium in game theory [77].

We model the game as a finite, zero-sum Partially Observed Stochastic Game (POSG) with one-sided partial observability:

$$\Gamma = \langle \mathcal{N}, \mathcal{S}, (\mathcal{A}_i)_{i\in\mathcal{N}}, \mathcal{T}, \mathcal{R}, \gamma, \rho_1, T, \mathcal{O}, \mathcal{Z} \rangle \tag{1}$$

It is a discrete-time game that starts at time $t = 1$ and ends at time $t = T$. 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. Hence, $\mathcal{N} = \{1, 2\}$.

Time horizon T. The time horizon T is a random variable that depends on both players' strategies and takes values in the set $T \in \{2, 3, \ldots, \infty\}$.

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

Action spaces $\mathcal{A}_i$. Each player $i \in \mathcal{N}$ can invoke two actions: "stop" (S) and "continue" (C). The action spaces are thus $\mathcal{A}_1 = \mathcal{A}_2 = \{S, C\}$. Executing action S triggers a change in the game while action C is a passive action. In the following, we encode S with 1 and C with 0.

The attacker can invoke the stop action twice: the first time to start the intrusion and the second time to terminate it.

The defender can invoke the stop action $L \geq 1$ times. A stop action is a defensive action against a possible intrusion. The number of stop actions remaining to the defender is known to both players and is denoted by $l \in \{1, \ldots, L\}$.

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

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

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

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

Transition probabilities $\mathcal{T}$. At each time-step t, a state transition from s_t to s_{t+1} occurs with probability $\mathcal{T}\big(s_{t+1}, s_t, (a_t^{(1)}, a_t^{(2)})\big) = \mathbb{P}\big[s_{t+1} \big| s_t, (a_t^{(1)}, a_t^{(2)})\big]$:

$$\mathcal{T}_{l>1}\big(0, 0, (S, C)\big) = \mathcal{T}\big(0, 0, (C, C)\big) = 1 \tag{2}$$

$$\mathcal{T}_{l>1}\big(1, 1, (\cdot, C)\big) = \mathcal{T}\big(1, 1, (C, C)\big) = 1 - \phi_l \tag{3}$$

$$\mathcal{T}_{l>1}\big(1, 0, (\cdot, S)\big) = \mathcal{T}\big(1, 0, (C, S)\big) = 1 \tag{4}$$

$$\mathcal{T}_{l>1}\big(\emptyset, 1, (\cdot, C)\big) = \mathcal{T}\big(\emptyset, 1, (C, C)\big) = \phi_l \tag{5}$$

$$\mathcal{T}_{l=1}\big(\emptyset, \cdot, (S, \cdot)\big) = \mathcal{T}(\emptyset, \emptyset, \cdot) = \mathcal{T}(\emptyset, 1, (\cdot, S)) = 1 \tag{6}$$

where $\mathcal{T}_{l>1}$ and $\mathcal{T}_{l=1}$ refer to the transition probabilities when $l > 1$ and $l = 1$, respectively. All other state transitions have probability 0.

(2)-(3) define the probabilities of the recurrent state transitions $0 \to 0$ and $1 \to 1$. 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 $l_t - a_t^{(1)} > 0$. Here ϕ_l denotes the probability that the intrusion is stopped, which is a parameter of the use case. The intrusion can be stopped at any time-step as a consequence of previous stop actions by the defender. (We assume ϕ_l increases with each stop action that the defender takes.)

(4) captures the transition $0 \to 1$, which occurs when the attacker chooses action S and $l_t - a_t^{(1)} > 0$. (5)-(6) 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. when $l_t - a_t^{(1)} = 0$); (*ii*) when the intrusion is stopped by the defender with probability ϕ_l; and *(iii)*, when $s_t = 1$ and the attacker terminates the intrusion ($a_t^{(2)} = 1$).

The evolution of the game can be described with the state transition diagram in Fig. 3. The figure captures a game *episode*, which starts at $t = 1$ and ends at $t = T$.

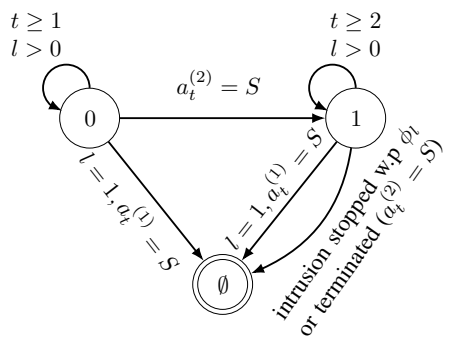

Fig. 3: State transition diagram of a game episode: each disk represents a state; an arrow represents a state transition; a label indicates the conditions for the state transition (w.p means "with probability"); a game episode starts in state $s_1 = 0$ with $l = L$ and ends in state $s_T = \emptyset$.

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

$$\mathcal{R}(\emptyset, \cdot) = 0 \tag{7}$$
$$\mathcal{R}\big(1, (\cdot, S)\big) = 0 \tag{8}$$
$$\mathcal{R}\big(0, (C, \cdot)\big) = 0 \tag{9}$$
$$\mathcal{R}\big(0, (S, \cdot)\big) = R_{cost}/l_t \tag{10}$$
$$\mathcal{R}\big(1, (S, C)\big) = R_{st}/l_t \tag{11}$$
$$\mathcal{R}\big(1, (C, C)\big) = R_{int} \tag{12}$$

(7)-(8) state that the reward is zero in the terminal state and when the attacker terminates an intrusion. (9) states that the defender incurs no cost when no attack occurs and it does not take a defensive action. (10) indicates that the defender incurs a cost when taking a defensive action if no intrusion is ongoing. (11) states that the defender receives a reward when taking a stop action while an intrusion occurs. Lastly, (12) indicates that the defender incurs a cost for each time-step during which an intrusion occurs.

Observation function $\mathcal{Z}$. At 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 $\mathcal{Z}(o_t, s_t, (a_{t-1}^{(1)}, a_{t-1}^{(2)}))= \mathbb{P}[o_t | s_t, (a_{t-1}^{(1)}, a_{t-1}^{(2)})]$ as follows:

$$\mathcal{Z}\big(o_t, 0, \cdot\big) = f_O(o_t|0) \tag{13}$$
$$\mathcal{Z}\big(o_t, 1, \cdot\big) = f_O(o_t|1) \tag{14}$$

Belief update. At time-step $t > 1$, the defender updates the belief state b_{t-1} using the equation:

$$b_t(s_t) = C \sum_{s_{t-1} \in \mathcal{S}} \sum_{a_{t-1}^{(2)} \in \mathcal{A}_2} b_{t-1}(s_{t-1}) \pi_2(a_{t-1}^{(2)} | s_{t-1}, b_{t-1}) \cdot$$
$$\mathcal{Z}(o_t, s_t, (a_{t-1}^{(1)}, a_{t-1}^{(2)})) \mathcal{T}\big(s_t, s_{t-1}, (a_{t-1}^{(1)}, a_{t-1}^{(2)})\big) \tag{15}$$

where $C = 1/\mathbb{P}[o_t | a_{t-1}^{(1)}, \pi_2, b_{t-1}]$ is a normalizing factor that makes the components of b_t sum to 1. The initial belief is $b_1(0) = 1$.

Player strategies π_i. A defender strategy is a function $\pi_1 \in \Pi_1 : \{1, \ldots, L\} \times \mathcal{B} \to \Delta(\mathcal{A}_1)$, where $\Delta(\mathcal{A}_1)$ denotes the set of probability distributions over $\mathcal{A}_1$. Similarly, an attacker strategy is a function $\pi_2 \in \Pi_2 : \{1, \ldots, L\} \times \mathcal{S} \times \mathcal{B} \to \Delta(\mathcal{A}_2)$. The strategies for both players are dependent on l but independent of t (i.e. strategies are stationary). If π_i always maps on to an action with probability 1, it is called *pure*, otherwise it is called *mixed*. In other words, a pure strategy is deterministic and a mixed strategy is stochastic.

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

$$J_1(\pi_1, \pi_2) = \mathbb{E}_{(\pi_1, \pi_2)} \left[\sum_{t=1}^{T} \gamma^{t-1} \mathcal{R}(s_t, \boldsymbol{a}_t) \right] \tag{16}$$
$$J_2(\pi_1, \pi_2) = -J_1(\pi_1, \pi_2) \tag{17}$$

where $\gamma \in [0, 1)$ is the discount factor and $\mathbb{E}_{(\pi_1, \pi_2)}$ denotes the expectation under strategy profile (π_1, π_2).

Best response strategies $\tilde{\pi}_i$. A defender strategy $\tilde{\pi}_1 \in \Pi_1$ is called a *best response* against $\pi_2 \in \Pi_2$ if it *maximizes* J_1 (16). Similarly, an attacker strategy $\tilde{\pi}_2$ is called a best response against π_1 if it *minimizes* J_1 (17). Hence, the best response correspondences B_1 and B_2 are obtained as follows:

$$B_1(\pi_2) = \arg\max_{\pi_1 \in \Pi_1} J_1(\pi_1, \pi_2) \tag{18}$$
$$B_2(\pi_1) = \arg\min_{\pi_2 \in \Pi_2} J_1(\pi_1, \pi_2) \tag{19}$$

Optimal strategies π_i^*. An optimal defender strategy π_1^* is a best response strategy against any attacker strategy that *minimizes* J_1. Similarly, an optimal attacker strategy π_2^* 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:

$$(\pi_1^*, \pi_2^*) \in B_1(\pi_2^*) \times B_2(\pi_1^*) \tag{20}$$

Since no player has an incentive to change its strategy, (π_1^*, π_2^*) is a Nash equilibrium [77].

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

Finding optimal strategies that satisfy (20) is equivalent to finding a Nash equilibrium for the POSG Γ (1). We know from game theory that Γ has at least one mixed Nash equilibrium [77]–[80]. (A Nash equilibrium is called mixed if one or more players follow mixed strategies.) In this section, we first analyze the structure of Nash equilibria in Γ using optimal stopping theory and then we describe an efficient reinforcement learning algorithm for approximating these equilibria.

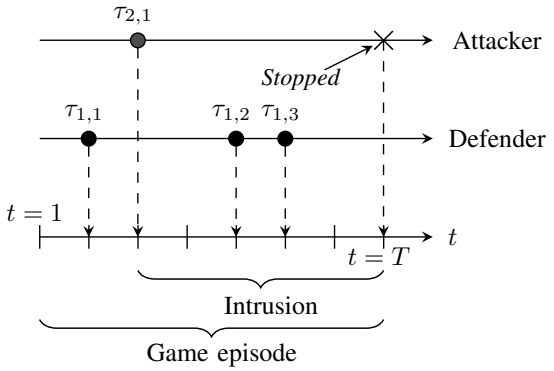

Fig. 4: Stopping times of the defender and the attacker in a game episode; the bottom horizontal axis represents time; the black circles on the middle axis and the upper axis represent time-steps of the defender's stop actions and the attacker's stop actions, respectively; $\tau_{i,j}$ denotes the jth stopping time of player i; the cross shows the time the intrusion is stopped; an intrusion starts when the attacker takes the first stop action (at time $\tau_{2,1}$); an episode ends either when the attacker is stopped (as a consequence of defender actions) or when the attacker terminates its intrusion.

A. Analyzing Best Responses using Optimal Stopping Theory

The equilibria in Γ can be obtained by finding the pairs of strategies that are best responses against each other (20). 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 $\mathcal{M}$. The corresponding Bellman equations are [81]:

$$V^*_{l,\pi_2}(b_t) = \max_{a^{(1)}_t \in \mathcal{A}_1} \mathop{\mathbb{E}}_{\pi_2, b_t, a^{(1)}_t} \left[r_{t+1} + \gamma V^*_{l-a^{(1)}_t, \pi_2}(b_{t+1}) \right] \quad (21)$$

$$V^*_{l,\pi_1}((b_t, s_t)) = \min_{a^{(2)}_t \in \mathcal{A}_2} \mathop{\mathbb{E}}_{\pi_1, a^{(2)}_t} \left[r_{t+1} + \gamma V^*_{l-a^{(1)}_t, \pi_1}((b_{t+1}, s_{t+1})) \right] \quad (22)$$

where V^*_{l,π_2} is the value function in the POMDP $\mathcal{M}^P$ given that the attacker follows strategy π_2 and the defender has l stops remaining, and V^*_{l,π_1} is the value function in the MDP $\mathcal{M}$ given that the defender follows strategy π_1 and has l stops remaining.

Since the game is zero-sum, stationary, and $\gamma < 1$, it follows from the Minimax theorem in game theory that there exists a value function:

$$V^*_l(b_t) = \max_{\pi_1 \in \Delta(\mathcal{A}_1)} \min_{\pi_2 \in \Delta(\mathcal{A}_2)} \mathbb{E}_{\pi_1, \pi_2, b_t} \left[r_{t+1} + \gamma V^*_{l-a^{(1)}_t}(b_{t+1}) \right] \quad (23)$$

and that $V^*_l(b) = V^*_{1,l,\pi^*_2}(b) = V^*_{2,l,\pi^*_1}(b, s)$ [78], [82]. Further, from Markov decision theory we know that for any strategy pair (π_1, π_2), a corresponding pair of *pure* best response strategies $(\tilde{\pi}_1, \tilde{\pi}_2) \in B_1(\pi_2) \times B_2(\pi_1)$ exists [83].

We interpret the POMDP $\mathcal{M}^P$ and the MDP $\mathcal{M}$ that determine the best response strategies as *optimal stopping* problems (see Fig. 4) [9], [56], [84], [85]. Consequently, an optimal solution to $\mathcal{M}^P$ (or $\mathcal{M}$) is also an optimal solution to the corresponding stopping problem and vice versa.

The problem for the defender is to find a stopping strategy $\pi^*_1(b_t) \to \{S, C\}$ that maximizes J_1 (16) and prescribes the optimal stopping times $\tau^*_{1,1}, \tau^*_{1,2}, \ldots, \tau^*_1$. Similarly, the problem for the attacker is to find a stopping strategy $\pi^*_2(s_t, b_t) \to \{S, C\}$ that minimizes J_1 (17) and prescribes the optimal stopping times $\tau^*_{2,1}$ and $\tau^*_{2,2}$.

Given a pair of stopping strategies (π_1, π_2) and their (pure) best responses $\tilde{\pi}_1 \in B_1(\pi_2)$ and $\tilde{\pi}_2 \in B_2(\pi_1)$, we define two subsets of $\mathcal{B} = [0, 1]$: the *stopping sets* and the *continuation sets*.

The stopping sets $\mathscr{S}^{(1)}$ and $\mathscr{S}^{(2)}$ include the belief states where S is a best response:

$$\mathscr{S}^{(1)}_{l,\pi_2} = \{b(1) \in [0, 1] : \tilde{\pi}_{1,l}\big(b(1)\big) = S\} \quad (24)$$

$$\mathscr{S}^{(2)}_{s,l,\pi_1} = \{b(1) \in [0, 1] : \tilde{\pi}_{2,l}\big(s, b(1)\big) = S\} \quad (25)$$

Similarly, the continuation sets $\mathscr{C}^{(1)}$ and $\mathscr{C}^{(2)}$ contain the belief states where C is a best response:

$$\mathscr{C}^{(1)}_{l,\pi_2} = \{b(1) \in [0, 1] : \tilde{\pi}_{1,l}\big(b(1)\big) = C\} \quad (26)$$

$$\mathscr{C}^{(2)}_{s,l,\pi_1} = \{b(1) \in [0, 1] : \tilde{\pi}_{2,l}\big(s, b(1)\big) = C\} \quad (27)$$

Based on [9], [82], [86]–[88], we formulate Theorem 1, which contains an existence result for equilibria and a structural result for best response strategies of the game.

Theorem 1. *Given the POSG Γ (1) with one-sided partial observability and $L \geq 1$, the following holds:*

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

(B) We assume that the probability mass function $f_{O|s}$ is totally positive of order 2 (i.e., TP2 [86, Definition 10.2.1, pp. 223]). Given an attacker strategy $\pi_2 \in \Pi_2$, then there exist values $\tilde{\alpha}_1 \geq \tilde{\alpha}_2 \geq \ldots \geq \tilde{\alpha}_L \in [0, 1]$ and a best response strategy $\tilde{\pi}_1 \in B_1(\pi_2)$ for the defender that satisfies

$$\tilde{\pi}_{1,l}(b(1)) = S \iff b(1) \geq \tilde{\alpha}_l \quad l \in \{1, \ldots, L\} \quad (28)$$

(C) Given a defender strategy $\pi_1 \in \Pi_1$ where $\pi_1(S|b(1))$ is non-decreasing in $b(1)$ and $\pi_1(S|1) = 1$, then there exist values $\tilde{\beta}_{0,1}$, $\tilde{\beta}_{1,1}$, …, $\tilde{\beta}_{0,L}$, $\tilde{\beta}_{1,L} \in [0, 1]$ and a best response strategy $\tilde{\pi}_2 \in B_2(\pi_1)$ for the attacker that satisfies

$$\tilde{\pi}_{2,l}(0, b(1)) = C \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{0,l} \quad (29)$$

$$\tilde{\pi}_{2,l}(1, b(1)) = S \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{1,l} \quad (30)$$

for $l \in \{1, \ldots, L\}$.

Proof. See Appendix A. □

Theorem 1 tells us that Γ has a mixed Nash equilibrium. Further, under assumptions generally met in practice, the best response strategies have threshold properties (see Fig. 5). In the following, we describe an algorithm that leverages these properties to efficiently approximate Nash equilibria of Γ.

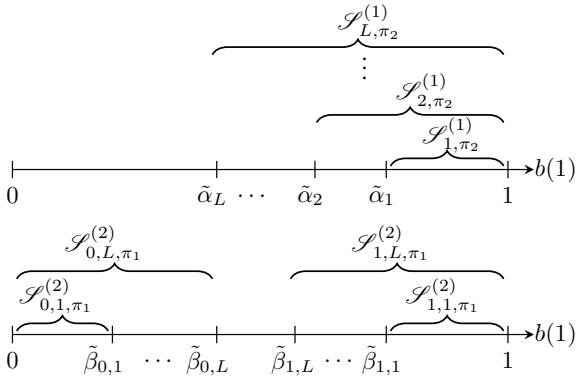

Fig. 5: Illustration of Theorem 1; the upper part shows L thresholds $\tilde{\alpha}_1 \geq \tilde{\alpha}_2, \ldots, \geq \tilde{\alpha}_L \in [0,1]$ that define a best response strategy $\tilde{\pi}_{1,\tilde{\theta}^{(1)}} \in B_1(\pi_2)$ for the defender (28); the lower part shows $2L$ thresholds $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$ that define a best response strategy $\tilde{\pi}_{2,\tilde{\theta}^{(2)}} \in B_2(\pi_1)$ for the attacker (29)-(30).

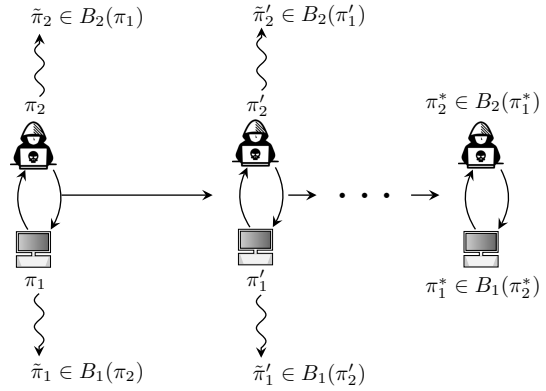

Fig. 6: The fictitious self-play process; in every iteration step each player learns a best response strategy $\tilde{\pi}_i \in B_i(\pi_{-i})$ and updates its strategy based on the empirical distribution of its past best response strategies; the horizontal arrows indicate iteration steps of self-play and the vertical arrows indicate the learning of best response strategies; The process converges towards a Nash equilibrium (π^*_1, π^*_2).

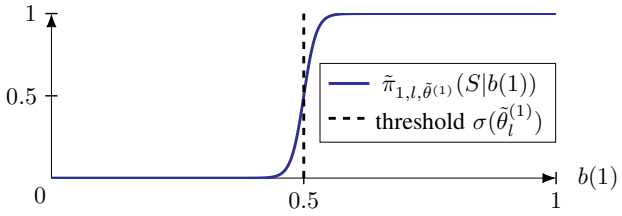

Fig. 7: A mixed threshold strategy where $\sigma(\tilde{\theta}^{(1)}_l)$ is the threshold (0.5 in this example); the x-axis indicates the defender's belief state $b(1) \in [0,1]$ and the y-axis indicates the probability prescribed by $\tilde{\pi}_{1,\tilde{\theta}^{(1)}}$ to the stop action S.

B. Finding Nash Equilibria through Fictitious Self-Play

Computing Nash equilibria for a POSG is generally intractable [82]. However, approximate solutions can be obtained through iterative methods. One such method is *fictitious self-play*, where both players start from random strategies and continuously update their strategies based on outcomes of played game episodes [89].

Fictitious self-play evolves through a sequence of iteration steps, which is illustrated in Fig. 6. An iteration step includes three stages. First, player 1 learns a best response strategy against player 2's current strategy. The roles are then reversed and player 2 learns a best response strategy against player 1's current strategy. Lastly, each player adopts a new strategy, which is determined by the empirical distribution over its past best response strategies. The sequence of iteration steps continues until the strategies of both players have sufficiently converged to a Nash equilibrium [89], [90].

C. Our Self-Play Algorithm: T-FP

We present a fictitious self-play algorithm called Threshold Fictitious Self-Play (T-FP), which efficiently approximates a Nash equilibrium of Γ based on Theorem 1. The pseudocode of T-FP is listed in Algorithm 1.

T-FP implements the fictitious self-play process described above and generates a sequence of strategy profiles (π_1, π_2), (π'_1, π'_2), ... that converges to a Nash equilibrium (π^*_1, π^*_2) [90]. During each step of this process, T-FP learns best responses against the players' current strategies and then updates the strategies of both players (see Fig. 6).

To learn the best response strategies $\tilde{\pi}_1 \in B_1(\pi_2)$ and $\tilde{\pi}_2 \in B_2(\pi_1)$, T-FP parameterizes $\tilde{\pi}_1$ and $\tilde{\pi}_2$ through threshold vectors according to Theorem 1. The defender's best response strategy $\tilde{\pi}_1$ is parameterized with the vector $\tilde{\theta}^{(1)} \in \mathbb{R}^L$ (32). Similarly, the attacker's best response strategy $\tilde{\pi}_2$ is parameterized with the vector $\tilde{\theta}^{(2)} \in \mathbb{R}^{2L}$ (33).

$$\varphi(a,b) \triangleq \left(1 + \left(\frac{b(1-\sigma(a))}{\sigma(a)(1-b)}\right)^{-20}\right)^{-1} \tag{31}$$

$$\tilde{\pi}_{1,\tilde{\theta}^{(1)}}\big(S|b(1)\big) \triangleq \varphi\left(\tilde{\theta}^{(1)}_l, b(1)\right) \tag{32}$$

$$\tilde{\pi}_{2,\tilde{\theta}^{(2)}}\big(S|b(1),s\big) \triangleq \varphi\left(\tilde{\theta}^{(2)}_{sL+l}, \pi_1(S|b(1))\right) \tag{33}$$

The parameterized strategies defined by (31)-(33) are mixed (and differentiable) strategies that approximate threshold strategies (see Fig. 7). In (31)-(33) $\sigma(\cdot)$ is the sigmoid function, $a \in \mathbb{R}$, $b \in \mathbb{R}$, $\sigma(\tilde{\theta}^{(1)}_1)$, $\sigma(\tilde{\theta}^{(1)}_2)$, ..., $\sigma(\tilde{\theta}^{(1)}_L) \in [0,1]$ 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 $\tilde{\pi}_{1,\tilde{\theta}^{(1)}}$ and $\tilde{\pi}_{2,\tilde{\theta}^{(2)}}$ by iteratively updating the threshold vectors $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic approximation. To update the threshold vectors, T-FP simulates Γ, which allows to evaluate the objective functions $J_1(\tilde{\pi}_{1,\tilde{\theta}^{(1)}}, \pi_2)$ and $J_2(\pi_1, \tilde{\pi}_{2,\tilde{\theta}^{(2)}})$ (16)-(17). The obtained values of J_1 and J_2

are then used to estimate the gradients $\nabla_{\tilde{\theta}^{(1)}} J_1$ and $\nabla_{\tilde{\theta}^{(2)}} J_2$ using the Simultaneous Perturbation Stochastic Approximation (SPSA) gradient estimator (lines 10-19 in Algorithm 1) [91], [92]. Next, the estimated gradients are used to update $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic gradient ascent (line 20). This procedure of estimating gradients and updating $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ continues for a given number of iterations (lines 9-21). Then, the threshold vectors $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ are added to buffers $\Theta^{(1)}$ and $\Theta^{(2)}$, which contain the vectors learned in previous iterations of T-FP (line 22). Finally, the T-FP iteration step is completed by having both players update their strategies based on the empirical distributions over the past vectors in the buffers (lines 24-25). The sequence of iteration steps continues until the strategies have sufficiently converged to a Nash equilibrium (lines 6-27).

(In Algorithm 1, $\mathcal{U}_k(\{-1,1\})$ denotes a k-dimensional discrete multivariate uniform distribution on $\{-1,1\}$ and π_{-i} denotes the strategy of player $j \in \mathcal{N} \setminus \{i\}$.)

Algorithm 1 T-FP: Threshold Fictitious Self-Play

Input
Γ, N: the POSG and # best response iterations
$a, c, \lambda, A, \epsilon, \delta$: scalar coefficients
Output
(π_1^*, π_2^*): an approximate Nash equilibrium

1: **procedure** T-FP
2: $\tilde{\theta}^{(1)} \sim \mathcal{U}_L(\{-1,1\}), \quad \tilde{\theta}^{(2)} \sim \mathcal{U}_{2L}(\{-1,1\})$
3: $\Theta^{(1)} \leftarrow \{\tilde{\theta}^{(1)}\}, \quad \Theta^{(2)} \leftarrow \{\tilde{\theta}^{(2)}\}, \quad \hat{\delta} \leftarrow \infty$
4: $\pi_1 \leftarrow$ EMPIRICALDISTRIBUTION($\Theta^{(1)}$)
5: $\pi_2 \leftarrow$ EMPIRICALDISTRIBUTION($\Theta^{(2)}$)
6: **while** $\hat{\delta} \geq \delta$ **do**
7: **for** $i \in \{1,2\}$ **do**
8: $\tilde{\theta}^{(i)}_{(1)} \sim \mathcal{U}_{iL}(\{-1,1\})$
9: **for** $n \in \{1, \ldots, N\}$ **do**
10: $a_n \leftarrow \frac{a}{(n+A)^\epsilon}, \quad c_n \leftarrow \frac{c}{n^\lambda}$
11: **for** $k \in \{1, \ldots, iL$ **do**
12: $(\Delta_n)_k \sim \mathcal{U}_1(\{-1,1\})$
13: **end for**
14: $R_{high} \sim J_i(\pi_{i,\tilde{\theta}^{(i)}_{(n)}} + c_n \Delta_n, \pi_{-i})$
15: $R_{low} \sim J_i(\pi_{i,\tilde{\theta}^{(i)}_{(n)}} - c_n \Delta_n, \pi_{-i})$
16: **for** $k \in \{1, \ldots, iL\}$ **do**
17: $G \leftarrow \frac{R_{high} - R_{low}}{2c_n (\Delta_n)_k}$
18: $\left(\hat{\nabla}_{\tilde{\theta}^{(i)}_{(n)}} J_i(\pi_{i,\tilde{\theta}^{(i)}_{(n)}}, \pi_{-i})\right)_k \leftarrow G$
19: **end for**
20: $\tilde{\theta}^{(i)}_{(n+1)} = \tilde{\theta}^{(i)}_{(n)} + a_n \hat{\nabla}_{\tilde{\theta}^{(i)}_{(n)}} J_i(\pi_{i,\tilde{\theta}^{(i)}_{(n)}}, \pi_{-i})$
21: **end for**
22: $\Theta^{(i)} \leftarrow \Theta^{(i)} \cup \tilde{\theta}^{(i)}_{(N+1)}$
23: **end for**
24: $\pi_1 \leftarrow$ EMPIRICALDISTRIBUTION($\Theta^{(1)}$)
25: $\pi_2 \leftarrow$ EMPIRICALDISTRIBUTION($\Theta^{(2)}$)
26: $\hat{\delta} =$ EXPLOITABILITY(π_1, π_2)
27: **end while**
28: **return** (π_1, π_2)
29: **end procedure**

V. EMULATING THE TARGET INFRASTRUCTURE TO INSTANTIATE THE SIMULATION AND TO EVALUATE LEARNED STRATEGIES

The T-FP algorithm described above approximates a Nash equilibrium of Γ by simulating game episodes and updating both players' strategies through stochastic approximation. T-FP requires the observation distribution conditioned on the system state $f_{O|s}$ (13)-(14). The emulation system shown in Fig. 2 allows us to estimate this distribution and later to evaluate the learned strategies.

This section describes the emulation system, our method for estimating $f_{O|s}$, and our method for evaluating defender strategies.

A. Emulating the Target Infrastructure

The emulation system executes on a cluster of machines that runs a virtualization layer provided by Docker containers and virtual links [93]. The system implements network isolation and traffic shaping using network namespaces and the NetEm module in the Linux kernel [94]. Resource allocation to containers, e.g. CPU and memory, is enforced using cgroups.

The network topology of the emulated infrastructure is shown in Fig. 1 and its configuration is given in Appendix C. The emulation system includes the clients, the attacker, the defender, network connectivity, and 31 devices of the target infrastructure (e.g. application servers and the gateway). The software functions on the emulation system replicate important components of the target infrastructure, such as, web servers, databases, and the Snort IDPS, which is deployed using Snort's community ruleset v2.9.17.1.

We emulate connections between servers as full-duplex lossless connections of 1 Gbit/s capacity in both directions. We emulate connections between the gateway and the external 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 based on measurements on enterprise and wide-area networks [95]–[97].)

B. Emulating the Client Population

The *client population* is emulated by processes in Docker containers. Clients interact with application servers through the gateway by performing a sequence of functions on a sequence of servers, both of which are selected uniformly at random from Table 1. Client arrivals per time-step are emulated using a stationary Poisson process with parameter $\lambda = 20$ and exponentially distributed service times with parameter $\mu = \frac{1}{4}$, which results in an average client-lifetime of 4 time-steps. The duration of a time-step is 30 seconds.

C. Emulating Defender and Attacker Actions

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

The defender executes either a continue action or a stop action. A continue action is virtual in the sense that it does not

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 invokes functions on application servers.

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

TABLE 2: Defender commands executed on the emulation system.

trigger any function in the emulation. A stop action, however, invokes specific functions in the emulated infrastructure. We have implemented $L = 7$ stop actions for the defender, which are listed in Table 2. The first stop action revokes user certificates and recovers user accounts expected to be compromised by the attacker. The second stop action updates the firewall configuration of the gateway to drop traffic from IP addresses flagged by the IDPS. Stop actions $3-6$ trigger the dropping of traffic that generates IDPS alerts of priorities $1-4$. The final stop action blocks all incoming traffic. (Note that according to Snort's terminology, 1 is the highest priority. We inverse the labeling in our framework for convenience.)

Like the defender, the attacker executes either a stop action or a continue action during each time-step. The attacker can only take two stop actions during a game episode. The first determines when the intrusion starts and the second when it terminates (see Section III).

During a game episode, the attacker executes a sequence of commands, drawn randomly from all of the commands

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 executed on the emulation system; exploits are identified according to their identifier in the Common Vulnerabilities and Exposures (CVE) database [98].

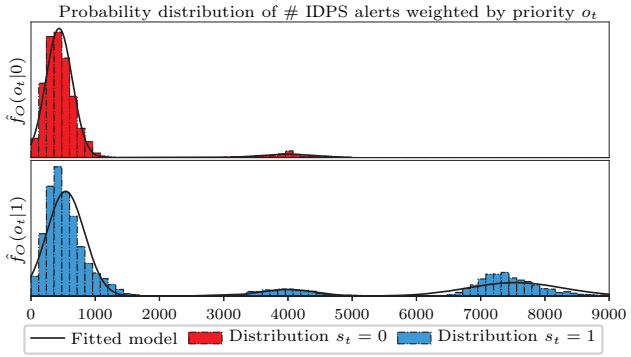

Fig. 8: Empirical distributions of o_t when no intrusion occurs ($s_t = 0$) and during intrusion ($s_t = 1$); the black lines show the fitted Gaussian mixture models.

listed in Table 3. The first in this sequence is executed when the attacker takes the first stop action. A further command is invoked whenever the attacker takes a continue action.

D. Estimating the IDPS Alert Distribution

At the end of every time-step, the emulation system collects the number of IDPS alerts with priorities 1-4 that occurred during the time-step. These values are then used to compute the metric o_t, which contains the total number of IDPS alerts, weighted by priority.

For the evaluation reported in this paper we collect measurements from $23,000$ time-steps.

Using these measurements, we apply expectation-maximization [99] to fit Gaussian mixture distributions $\hat{f}_{O|0}$ and $\hat{f}_{O|1}$ as estimates of $f_{O|0}$ and $f_{O|1}$ (13)-(14).

Fig. 8 shows the empirical distributions and the fitted models 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 $0 - 1000$. $\hat{f}_{O|1}$ 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, which are almost all non-negative. This suggests to us that the TP2 assumption in Theorem 1 can be made.

E. Running a Game Episode

During a game episode, the state evolves according to the dynamics defined by (2)-(6), the defender's belief state evolves according to (15), the players' rewards are calculated using the reward function $\mathcal{R}$ (7)-(12), the defender's observations are obtained from f_O (13)-(14), and the actions of both players are determined by their respective strategies. If the game runs in the emulation system, the player's actions include executing networking and computing functions (see Tables 2-3), and the observations from f_O are obtained through reading log files and metrics of the emulated infrastructure. (To collect the logs and system metrics from the emulation, we run software sensors that write to a distributed queue implemented with Kafka [100].) In the case of a game in the

simulation system, the observations are instead sampled from the estimated distribution $\hat{f}_O$.

VI. Learning Nash Equilibrium Strategies for the Target Infrastructure

Our approach to finding near-optimal defender strategies includes: (*i*) emulating the target infrastructure to obtain statistics for instantiating the simulation system; (*ii*) learning Nash equilibrium strategies using the T-FP algorithm in Section IV; and (*iii*), evaluating learned strategies on the emulation system in Section V (see Fig. 2). This section describes the learning process and the evaluation results of the intrusion response use case.

A. Learning Equilibrium Strategies through Self-Play

We run T-FP for 500 iteration steps to estimate a Nash equilibrium using the iterative method in Section IV-B, which is sufficient to meet the termination condition (line 6 in Algorithm 1). These iteration steps generate a sequence of strategy pairs $(\pi_1, \pi_2)_1$, $(\pi_1, \pi_2)_2$, ..., $(\pi_1, \pi_2)_{500}$.

At the end of each iteration step, we evaluate the current strategy pair (π_1, π_2) by running 500 evaluation episodes in the simulation system and 5 evaluation episodes in the emulation system. This process allows us to produce learning curves for different performance metrics (see Fig. 9).

The 500 training iterations and the associated evaluations constitute one *training run*. We run four training runs with different random seeds. A single training run takes about 5 hours of processing time in the simulation system. In addition, it takes around 12 hours to evaluate the strategies on the emulation system. The hyperparameters of T-FP are listed in Appendix B.

Computing environment for simulation and emulation. The environment for running simulations and training strategies is a Tesla P100 GPU.

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 response use case are available at [101]. They can be used to validate our results and to extend this research.

Convergence metric for T-FP. To estimate the convergence of the sequence of strategy pairs generated by T-FP, we use the *approximate exploitability* metric $\hat{\delta}$ [102]:

$$\hat{\delta} = J_1(\hat{\pi}_1, \pi_2) + J_2(\pi_1, \hat{\pi}_2) \tag{34}$$

where $\hat{\pi}_i$ denotes an approximate best response strategy for player i and the objective functions J_1 and J_2 are defined in (16) and (17), respectively. The closer $\hat{\delta}$ becomes to 0, the closer (π_1, π_2) is to a Nash equilibrium.

Baseline algorithms. We compare the performance of T-FP with that of two popular algorithms in previous work that use reinforcement learning and study use cases similar to ours [70], [82], [103]–[105]. The first algorithm is Neural Fictitious Self-Play (NFSP) [106], which is a general fictitious self-play algorithm that does not exploit the threshold structures expressed in Theorem 1. The second algorithm is Heuristic Search Value Iteration (HSVI) for one-sided POSGs [104], which is a state-of-the-art dynamic programming algorithm for one-sided POSGs.

Defender baseline strategies. We compare the dynamic defender strategies learned through T-FP with three static baseline strategies. The first baseline prescribes the stop action when an IDPS alert occurs, i.e., when $o_t > 0$. The second baseline is derived from the Snort IDPS, which is a de-facto industry standard and can be considered state-of-the-art for our use case. This baseline uses the Snort IDPS's recommendation system and takes a stop action when Snort has dropped 100 IP packets (see Appendix C for the Snort configuration). The third baseline assumes prior knowledge of the intrusion time and performs all L stops during the L subsequent time-steps.

Although a growing body of work uses reinforcement learning and game theory to find intrusion response strategies (see Section VII), a direct comparison between the defender strategies learned in our framework and those found in previous work is not feasible for two reasons. First, nearly all of the prior works have developed defender strategies for custom simulations [8], [10], [10], [11], [18]–[27], [33]–[35], [38], [39], [60], [62], [63], [67]–[72], [107]–[119] and there is no obvious way to map their solutions to an emulated environment like ours (see Fig. 1 and Appendix C). Second, the few prior works that study emulated infrastructures similar to ours either consider static attackers in fully observed environments [28]–[31], [36], [37], [48], [51], [120] or focus on use cases that are different from the one considered in this article [120], [121].

B. Evaluating the Learned Strategies

Fig. 9 shows the learning curves of the strategies obtained during the T-FP self-play process and the baselines introduced above. The red curve represents the results from the simulator; the blue curves show the results from the emulation system; the purple curves give the performance of the Snort IDPS baseline; the orange curves relate to the baseline strategy that mandates a stop action when an IDPS alert occurs; and the dashed black curve gives the performance of the baseline strategy that assumes prior knowledge of the intrusion time.

We note that all learning curves in Fig. 9 converge, which suggests that the learned strategies converge as well. (Fig. 9 only shows the first 120 iterations of the 500 iterations we performed, as the curves converge after 100 iterations.) Specifically, we observe that the approximate exploitability (34) of the learned strategies converges to small values (left plot), which indicates that the learned strategies approximate a Nash equilibrium both in the simulator and in the emulation system. Further, we see from the plot in the middle that both baseline strategies show decreasing performance as the attacker updates its strategy. In contrast, the defender strategy learned through T-FP improves its performance over time. This shows the benefit of a game-theoretic approach where the defender strategy is optimized against a dynamic attacker. Lastly, we notice that the average intrusion length of the learned defender strategy and the Snort IDPS baseline strategy is 2 and 3, respectively (right plot). In comparison, the average

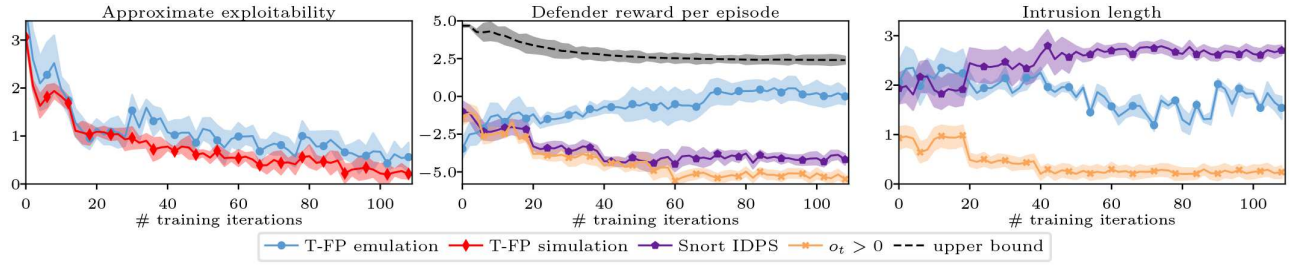

Fig. 9: Learning curves from the self-play process with T-FP; the red curve shows simulation results and the blue curves show emulation results; the purple, orange, and black curves relate to baseline strategies; the figures show different performance metrics: exploitability (34), episodic reward, and the length of intrusion; the curves indicate the mean and the 95% confidence interval over four training runs with different random seeds.

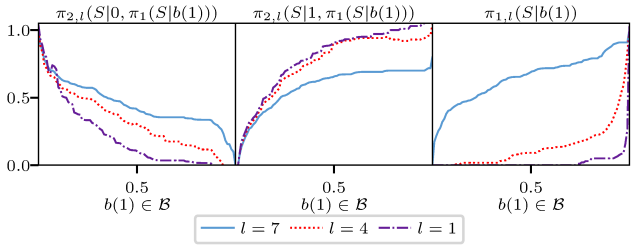

Fig. 10: Probability of the stop action S by the learned equilibrium strategies in function of $b(1)$ and l; the left and middle plots show the attacker's stopping probability when $s = 0$ and $s = 1$, respectively; the right plot shows the defender's stopping probability.

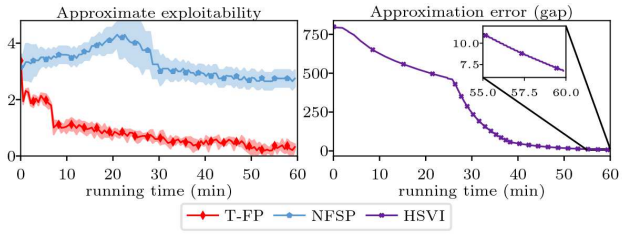

Fig. 11: Comparison between T-FP and two baseline algorithms: NFSP and HSVI; all curves show simulation results; the red curve relates to T-FP; the blue curve to NFSP; the purple curve to HSVI; the left plot shows the approximate exploitability metric (34) and the right plot shows the HSVI approximation error [104]; the curves depicting T-FP and NFSP show the mean and the 95% confidence interval over four training runs with different random seeds.

intrusion length of the baseline strategy $o_t > 0$ is close to 0, which indicates that it tends to prescribe all stop actions before an intrusion occurs.

Fig. 10 represents the strategies learned through T-FP in a simple form. The y-axis shows the probability of a stop action and the x-axis shows the defender's belief $b(1) \in \mathcal{B}$ that an intrusion occurs. The strategies are clearly stochastic. This is consistent with Theorem 1.A., which predicts a mixed Nash equilibrium. Further, Theorem 1.B predicts that the defender's stopping probability is increasing with $b(1)$ and decreasing with l, which is visible in the right plot. Similarly, Theorem 1.C predicts that the attacker's stopping probability decreases with the defender's stopping probability when $s = 0$ and increases when $s = 1$, which can be seen in the left and the middle plot.

Fig. 11 compares T-FP with the two baseline algorithms NFSP and HSVI on the simulator. NFSP implements fictitious self-play and can thus be compared with T-FP with respect to approximate exploitability (34). We observe in the left plot that T-FP converges much faster than NFSP. We explain the rapid convergence of T-FP by its design, which exploits structural properties of the stopping game.

The right plot shows that HSVI reaches an HSVI approximation error below 5 within an hour of processing time. Based on the recent literature we anticipated a much longer processing time [82], [122]. This suggests to us that T-FP and HSVI have similar convergence properties. A more detailed comparison between T-FP and HSVI is hard to perform due to the different nature of the two algorithms.

Fig. 12 shows the estimated value function of the game $\hat{V}_l^* : \mathcal{B} \to \mathbb{R}$ (23), where $\hat{V}_l^*(b(1))$ is the expected cumulative reward when the game starts in the belief state $b(1)$, the defender has l stops remaining, and both players follow optimal (equilibrium) strategies.

We see in Fig. 12 that $\hat{V}_l^*$ is piece-wise linear and convex, as expected from the theory of one-sided POSGs [122]. The figure indicates that $\hat{V}_l^*(b(1)) \leq 0$ for all $b(1) \in \mathcal{B}$ and that $\hat{V}_l^*(1) = 0$ for all $l \in \{1, \ldots, L\}$. Further, we note that the value of $\hat{V}_l^*$ is minimal when $b(1)$ is around 0.25 and that the values for $l = 1$ and $l = 7$ are very close.

That $\hat{V}_l^*(b(1)) \leq 0$ for all $b(1) \in \mathcal{B}$ and all $l \in \{1, \ldots, L\}$ has an intuitive explanation. For any $b(1)$, the attacker has the option to never attack if $s = 0$ or to abort an attack if $s = 1$. Both options yield a cumulative reward less than or equal to 0 (7)-(12). As a consequence, $\hat{V}_l^*(b(1)) \leq 0$ for any optimal attacker strategy and all $b(1) \in \mathcal{B}$ and $l \in \{1, \ldots, L\}$. (Recall that the attacker aims to minimize reward.)

The fact that $\hat{V}_l^*(b(1)) = 0$ when $b(1) = 1$ can be understood as follows. $b(1) = 1$ means that the defender knows that an intrusion occurs and will take defensive actions (see Theorem 1.B). Hence, when $b(1) = 1$, an attacker can only avoid detection by aborting the intrusion, which causes the game to end and yields a reward of zero, i.e. $\hat{V}_l^*(1) = 0$

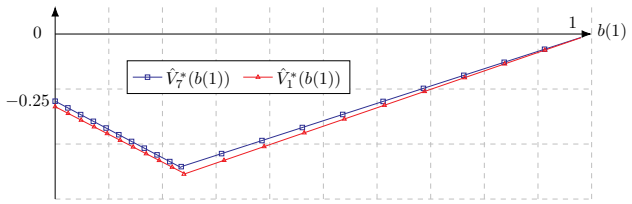

Fig. 12: The value function $\hat{V}_l^*(b(1))$ (23) computed through the HSVI algorithm with approximation error 4; the blue and red curves relate to $l = 7$ and $l = 1$, respectively.

for all $l \in \{1, \ldots, L\}$.

We interpret the fact that $\arg\min_{b(1)} \hat{V}_l^*(b(1))$ is around 0.25 as follows. The value of $b(1)$ that obtains the minimum corresponds to the belief state where the attacker achieves the lowest expected reward in the game. Negative rewards in the game are obtained when the defender mistakes an intrusion for no intrusion and vice versa (7)-(12). As a consequence, the attacker prefers belief states where the defender has a high uncertainty, e.g. $b(1) = 0.5$. At the same time, the attacker does not want $b(1)$ to be so large that the defender performs all its defensive actions before it gets a chance to attack, which can explain why we find the minimum to be around 0.25 rather than 0.5.

C. Discussion of the Evaluation Results

In this work, we propose a framework for analyzing and solving the intrusion response use case, which we validate both theoretically and experimentally. The key findings can be summarized as follows:

(*i*) Our framework is able to efficiently approximate optimal defender strategies for a practical IT infrastructure (see Fig. 9). While we have not evaluated the learned strategies in the target infrastructure due to safety reasons, the fact that they achieve almost the same performance in the emulated infrastructure as in the simulator gives us confidence that the obtained strategies would perform as expected in the target infrastructure.

(*ii*) The theory of optimal stopping provides insight about optimal strategies for attackers and defenders, which enables efficient computation of near-optimal strategies through self-play reinforcement learning (see Fig. 11). This finding can be explained by the threshold structures of the optimal stopping strategies, which drastically reduce the search space of possible strategies (see Theorem 1 and Algorithm 1).

(*iii*) Static defender strategies' performance deteriorate against a dynamic attacker, whereas defender strategies obtained through T-FP improve over time (see the middle plot in Fig. 9). This finding is consistent with previous studies that use game-theoretic approaches (e.g. [66], [67]) and suggests limitations of static defense systems, such as the Snort IDPS.

VII. Related Work

Since the early 1990s, there has been a broad interest in automating network security functions, especially in the areas of intrusion detection, intrusion prevention, and intrusion response.

In the area of intrusion detection, the traditional approach has been to use packet inspection and static rules for detection of intrusions [1], [75], [123]. The main drawback of this approach lies in the need for domain experts to configure the rule sets. As a consequence, much effort has been devoted to developing statistical methods for detecting intrusions. Examples of statistical methods include anomaly detection methods (e.g. [124]), change-point detection methods (e.g. [125]), Bayesian methods (e.g. [126]), hidden Markov modeling (e.g. [127]), deep learning methods (e.g. [128], [129]), and threat intelligence methods (e.g. [14]). As a result of this effort, all mainstream IDSs today have statistical components for automated detection of intrusions [74], [75], [130]–[132].

In contrast to intrusion detection, intrusion prevention and response usually remains a manual process performed by network administrators. Current IPSs and IRSs can be configured with rules to automatically match response actions to known intrusion types, but they have no means to find effective response actions in an automatic way [73], [75], [133], [134]. The problem of automatically finding response actions is an active area of research that uses concepts and methods from various fields, most notably from reinforcement learning (see surveys [15], [16], [135] and textbook [136]), control theory (see surveys [3], [4], [137] and example [61]), causal modeling (see example [138]), game theory (see textbooks [7], [139]–[141]), graph theory (see examples [142], [143]), fuzz testing (see examples [144], [145]), formal synthesis (see example [146]), attack graphs (see example [134]), artificial intelligence (see surveys [147], [148] and textbook [149]), and evolutionary methods (see examples [5], [6]).

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 security strategies through automatic control and reinforcement learning.

A. Game-Theoretic Modeling in Network Security

Since the early 2000s, researchers have studied automated security through modeling attacks and response actions on an IT infrastructure as a game between an attacker and a defender (see textbooks [7], [139]–[141]). The game is modeled in different ways depending on the use case. Examples from the literature include: advanced persistent threat games [54], [66]–[68], [71], [108], [109], [150], honeypot placement games [112]–[114], resource allocation games [38], [151], authentication games [8], distributed denial-of-service games [115], [120], situational awareness games [152], [153], moving target defense games [107], [154], and intrusion response games [10], [28], [45], [50], [72], [111], [116]–[119], [121]. These games are formulated using various models from the game-theoretic literature. For example: stochastic games (see e.g. [8], [68], [116], [119]), extensive-form games (see e.g. [7], [118]), Blotto games (see e.g. [38]), differential games (see e.g. [110]), hypergames (see e.g. [108], [109]), POSGs (see e.g. [10], [28], [45], [50], [115]), Stackelberg games (see e.g. [115], [117], [121]), graph-based games (see e.g. [107], [151]), evolutionary games (see e.g. [110], [111]), continuous-kernel games (see

e.g. [118]), rivalry games (see e.g. [54]), and Bayesian games (see e.g. [154]).

This paper differs from the works referenced above in two main ways. First, we model the intrusion response 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. Second, we evaluate obtained strategies on an emulated IT infrastructure. This contrasts with most of the prior works that use game-theoretic approaches, which either evaluate strategies analytically or in simulation [8], [10], [54], [66]–[68], [71], [72], [107]–[119], [150].

Game-theoretic formulations based on optimal stopping theory can be found in prior research on Dynkin games [57], [155]–[158]. Compared to these articles, our approach is more general by (*i*) allowing each player to take multiple stop actions within an episode; and (*ii*), by not assuming a game of perfect information. Another difference is that the referenced articles 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 response.

Our stopping game has similarities with the FlipIt game [66] and signaling games [159], both of which are commonplace in the security literature (see survey [160] and textbooks [7], [139]–[141]). Signaling games have the same information asymmetry as our game and FlipIt uses the same binary state space to model the state of an attack. 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 response use case and is an asymmetric zero-sum game. Lastly, compared to signaling games, the main difference is that our game is a sequential and simultaneous-move game. Signaling games, in comparison, are typically two-stage games where one player moves in each stage.

Previous game-theoretic studies that use emulation systems similar to ours are [120] and [121]. In [120], a denial-of-service use case is formulated as a signaling game, for which a Nash equilibrium is derived. The equilibrium is then used to design a defense mechanism that is evaluated in a software-defined network emulation based on Mininet [161]. Compared to this paper, the main differences are that we focus on a different use case than [120] and that our solution method is based on reinforcement learning.

Similar to this paper, the authors of [121] formulate an intrusion response use case as a POSG where the defender observes alerts from a Snort IDPS [75]. In contrast to our approach, however, the approach of [121] assumes access to attack-defense trees designed by human experts. Another difference between this paper and [121] is the POSG. The POSG in [121] has a larger state space than the POSG considered in this paper. Although this makes the POSG in [121] more expressive than ours, it also makes computation of optimal defender strategies intractable. In fact, to estimate optimal defender strategies, the authors of [121] are forced to approximate their model with one that has a smaller state space and is fully observed. In comparison, we are able to efficiently approximate equilibria of our game, without relying on model simplifications and without assuming access to attack-defense trees designed by human experts.

B. Control Theory for Automated Intrusion Response

Control theory provides a well-established mathematical framework for studying automatic systems. Classical control systems involve actuators in the physical world (e.g. electric power systems [162]) and many studies have focused on applying control theory to automate intrusion responses in cyber-physical systems (see surveys [163]–[165]).

The control framework can also be applied to computing systems and interest in control theory among researchers in IT security is growing (see survey [4]). As opposed to classical control theory, which is focused on *continuous-time* systems, the research on applying control theory to computing systems is focused almost entirely on *discrete-time* systems. The main reason being that measurements from computer systems are solicited on a sampled basis, which is best described by a discrete-time model [137], [166].

Previous works that apply control theory to the use case of intrusion response include: [60]–[63], [167]–[169]. All of which model the problem of selecting response actions as the problem of controlling a discrete-time dynamical system and obtain optimal defender strategies through dynamic programming.

The main limitation of the works referenced above is that dynamic programming does not scale to problems of practical size due to the curse of dimensionality [170], [171].

C. Reinforcement Learning for Automated Intrusion Response

Reinforcement learning has emerged as a promising approach to approximate optimal control strategies in scenarios where exact dynamic programming is not applicable, and fundamental breakthroughs demonstrated by systems like AlphaGo in 2016 [172] and OpenAI Five in 2019 [173] have inspired us and other researchers to study reinforcement learning with the goal to automate security functions (see surveys [15], [16]).

A large number of studies have focused on applying reinforcement learning to use cases similar to the intrusion response use case we discuss in this paper [9]–[11], [17]–[52], [64], [72]. These works use a variety of models, including MDPs [20], [23], [25], [26], [31], [34], [36], [42], [51], [52], [64], Stochastic games [10], [18], [28], [33], [45], [72], attack graphs [35], Petri nets [43], and POMDPs [9], [11], [21], [27], as well as various reinforcement learning algorithms, including Q-learning [18], [20], [23], [40], [43], [48], [64], [69], SARSA [21], PPO [10], [11], [34], [35], [37], hierarchical reinforcement learning [25], DQN [26], [36]–[39], [45], [51], Thompson sampling [27], MuZero [28], NFQ [29], DDQN [31], [50], NFSP [70], [103], A2C [42], A3C [49], and DDPG [30], [33].

This paper differs from the works referenced above in three main ways. First, we model the intrusion response 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 the game-theoretic model is that it allows finding defender strategies that are effective against a dynamic attacker, i.e. an attacker that adapts its strategy in response to the defender strategy.

Second, in a novel approach, we derive structural properties of optimal defender strategies in the game using optimal stopping theory.

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 [10], [11], [18]–[27], [33]–[35], [38], [39], [44]–[46], [50], [52], [52], [64], [69], [70], [72] 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 learning of security strategies that make use of emulation are: [48], [28], [29], [30], [31], [36], [47], [49], [51], [53], and [37]. They either emulate software-defined networks based on Mininet [161] or use custom testbeds. The main differences between these efforts and the work described in this article are: (*i*) we develop our own emulation system which allows for experiments with a large variety of exploits; (*ii*) we focus on a different use case (most of the referenced works study denial-of-service attacks); (*iii*) we do not assume that the defender has perfect observability; (*iv*) we do not assume a static attacker; and (*v*), 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, [174], [175], and [176] describe efforts in building emulation platforms for reinforcement learning and cyber defense, which resemble our emulation system. In contrast to these articles, our emulation system has been built to investigate the specific use case of intrusion response and forms an integral part of our general solution method (see Fig. 2).

VIII. Conclusion and Future Work

In this work, we combine a formal framework with a practical evaluation to address the problem of automated intrusion response. We formulate the interaction between an attacker and a defender as an optimal stopping game. This formulation gives us insight into the structure of optimal strategies, which we prove to have threshold properties. Based on this knowledge, we develop a fictitious self-play algorithm, Threshold Fictitious Self-Play (T-FP), which learns near-optimal strategies in an efficient way. The results from running T-FP show that the learned strategies converge to an approximate Nash equilibrium and thus to near-optimal strategies (see Fig. 9). The results also demonstrate that T-FP converges faster than a state-of-the-art fictitious self-play algorithm by taking advantage of threshold properties of optimal strategies (see Fig. 11). The threshold properties further enable us to provide a graphic representation of the learned strategies in a simple form (see Fig. 10).

To assess the learned strategies in a real environment, we evaluate them in a system that emulates our target infrastructure (see Fig. 1). The results show that the strategies achieve almost the same performance in the emulated infrastructure as in the simulation. This gives us confidence that the obtained strategies would perform as expected in the target infrastructure, which is not feasible to evaluate directly.

We plan to continue this work in several directions. First, we will extend the current model of the attacker and defender, which captures the timing of actions, to include decisions about a range of attacker and defender actions. Second, we plan to combine the strategies learned through our framework with techniques for online play, such as rollout [177]. Third, we plan to study techniques that allow to obtain defender strategies that generalize to a variety of infrastructure configurations and topologies.

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
Proofs

A. Proof of Theorem 1.A

Proof. Since the POSG Γ in (1) is finite and $\gamma \in (0, 1)$, the existence proofs in [80] and [122] 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 (7)-(12) and (18) that the pure strategy defined by $\bar{\pi}_1(0) = C$ and $\bar{\pi}_1(b(1)) = S \iff b(1) > 0$ is a best response for the defender against any attacker strategy when $s = 0 \iff b(1) = 0$. Similarly, given $\bar{\pi}_1$, we conclude from (7)-(12) and (19) that the pure strategy defined by $\bar{\pi}_2(0, b(1)) = C$ and $\bar{\pi}_2(1, b(1)) = S$ for all $b(1) \in [0, 1]$ is a best response for the attacker. Hence, $(\bar{\pi}_1, \bar{\pi}_2)$ is a pure Nash equilibrium (20). □

B. Proof of Theorem 1.B.

Proof. Given the POSG Γ (1) and a fixed attacker strategy π_2, any best response strategy for the defender $\tilde{\pi}_1 \in B_1(\pi_2)$ 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 π_1^* in $\mathcal{M}^P$ that satisfies (28). Conditions for (28) to hold and an existence proof are given in our previous work [9][Theorem 1.C]. Since $f_{O|s}$ is TP2 by assumption and all of the remaining conditions hold by definition of Γ (1), the result follows. □

C. Proof of Theorem 1.C.

Given the POSG Γ (1) and a fixed defender strategy π_1, any best response strategy for the attacker $\tilde{\pi}_2 \in B_2(\pi_1)$ is an optimal strategy in an MDP $\mathcal{M}$ (see Section IV). Hence, it is sufficient to show that there exists an optimal strategy π_2^* in $\mathcal{M}$ that satisfies (29)-(30). To prove this, we use properties of $\mathcal{M}$'s value function $V^*_{\pi_1,l}$ (22).

We use the value iteration algorithm to establish properties of $V^*_{\pi_1,l}$ [83], [86]. Let $V^k_{\pi_1,l}$, $\mathscr{S}^{k,(2)}_{s,l,\pi_1}$, and $\mathscr{C}^{k,(2)}_{s,l,\pi_1}$, denote the value function, the stopping set (25), and the continuation set (27) at iteration k of the value iteration algorithm, respectively. Then, $\lim_{k\to\infty} V^k_{\pi_1,l} = V^*_{\pi_1,l}$, $\lim_{k\to\infty} \mathscr{S}^{k,(2)}_{s,l,\pi_1} = \mathscr{S}^{(2)}_{s,l,\pi_1}$, and $\lim_{k\to\infty} \mathscr{C}^{k,(2)}_{s,l,\pi_1} = \mathscr{C}^{(2)}_{s,l,\pi_1}$ [83], [86]. We define $V^0_{\pi_1,l}\big((s,b(1))\big) = 0$ 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 π_1, $V^*_{\pi_1,l}\big(s,b(1)\big) \geq 0$ for all $s \in \mathcal{S}$ and $b(1) \in [0,1]$.*

Proof. Consider $\bar{\pi}_2$ defined by $\bar{\pi}_2(0,\cdot) = C$ and $\bar{\pi}_2(1,\cdot) = S$. Then it follows from (7)-(12) that for any $\pi_1 \in \Pi_1$, any $s \in \mathcal{S}$, and any $b(1) \in [0,1]$, the following holds: $V^{\bar{\pi}_2}_{\pi_1,l}(s,b(1)) \geq 0$. By optimality, $V^{\bar{\pi}_2}_{\pi_1,l}(s,b(1)) \leq V^*_{\pi_1,l}(s,b(1))$. Hence, $V^*_{\pi_1,l}(s,b(1)) \geq 0$. □

Lemma 2. *$V^*_{\pi_1,l}\big(1,b(1)\big)$ is non-increasing with $\pi_1(S|b(1))$ and non-decreasing with $l \in \{1,\ldots,L\}$.*

Proof. We prove this statement by mathematical induction. For $k=1$, we know from (7)-(12) that $V^1_{\pi_1,l}\big(1,\, b(1)\big)$ is non-increasing with $\pi_1(S|b(1))$ and non-decreasing with l.

For $k>1$, $V^k_{\pi_1,l}$ is given by:

$$V^k_{\pi_1,l}\big(1,b(1)\big) = \max\Big[0, -R\big(1,(C,a^{(1)})\big) + (1-\phi_l)\sum_o f_O(o|1)V^{k-1}_{l-a^{(1)}}\big(1,b(1)\big)\Big] \tag{35}$$

The first term inside the maximization in (35) is trivially non-increasing with $\pi_1(S|b(1))$ and non-decreasing with l. Assume by induction that the statement of Lemma 2 holds for $V^{k-1}_{\pi_1,l}\big(s,\, b(1)\big)$. Then the second term inside the maximization in (35) is non-increasing with $\pi_1(S|b(1))$ and non-decreasing with l by (7)-(12) and the induction hypothesis. Hence, $V^k_{\pi_1,l}\big(s,\, b(1)\big)$ is non-increasing with $\pi_1(S|b(1))$ and non-decreasing with l for all $k \geq 0$. □

Lemma 3. *If f_O is TP2 and $\pi_1(S|b(1))$ is increasing with $b(1)$, then $V_{\pi_1,l}(b(1),1) \geq \sum_o f_O(o|1)V_{\pi_1,l}(1,b^o(1))$, where $b^o(1)$ denotes $b(1)$ updated with (15) after observing $o \in \mathcal{O}$.*

Proof. Since f_O is TP2, it follows from [86, Theorem 10.3.1, pp. 225 and 238] and [9, Lemma 4, pp. 12] that given two beliefs $b'(1) \geq b(1)$ and given two observations $o \geq \bar{o}$, the following holds for any $k \in \mathcal{O}$ and $l \in \{1,\ldots,L\}$: $b'^{,o}(1) \geq b^o(1)$, $\mathbb{P}[o \geq k|b'(1)] \geq \mathbb{P}[o \geq k|b(1)]$, and $b^o_a(1) \geq b^{\bar{o}}_a(1)$.

Since π_1 is increasing with $b(1)$ and $V_{\pi_1,l}(b(1),1)$ is decreasing with $b(1)$ (Lemma 2), it follows that $\mathbb{E}_o[b^o(1)] \geq b(1)$, and thus $V_{\pi_1,l}(b(1),1) \geq \sum_o f_O(o|1)V_{\pi_1,l}(1,b^o(1))$. □

Lemma 4. *If f_O is TP2, $\pi_1(S|b(1)) = 1$, and $\pi_1(S|b(1))$ is increasing with $b(1)$, then $V^*_{\pi_1,l}\big(s,b(1)\big) = 0$ and for any $\tilde{\pi}_2 \in B_2(\pi_1)$, $\tilde{\pi}_2(1,b(1)) = S$.*

Proof. From (21)-(23) we know that $\tilde{\pi}_2(1,b(1)) = S$ iff:

$$R_{st}/l + (\phi_l - 1)\sum_o f_O(o|1)V^*_{\pi_1,l-a^{(1)}}(1,b^o(1)) \geq 0 \tag{36}$$

We know that $R_{st} \geq 0$ (see Section III). Further, since f_O is TP2, $\pi_1(S|b(1)) = 1$, and since $\pi_1(S|b(1))$ is increasing with $b(1)$, we have by Lemma 3 that $\mathbb{E}_o[\pi_1(S|b^o(1))] = 1$. As a consequence, the second term in the left-hand side of (36) is zero. Hence, the inequality holds and $\tilde{\pi}_2(1,b(1)) = 1$, which implies that $V^*_{\pi_1,l}\big(s,b(1)\big) = 0$. □

Lemma 5. *Given any defender strategy $\pi_1 \in \Pi_1$, if $\pi_2^*(1,b(1)) = S$, then $\pi_2^*(0,b(1)) = C$.*

Proof. $\pi_2^*(1,b(1)) = S$ implies that $V^*_{\pi_1,l}(1,b(1)) = 0$. Hence, it follows from Lemma 3 that:

$$\begin{aligned}&(1-\phi_l)\sum_{o\in\mathcal{O}} f_O(o|1)V^*_{\pi_1,l}(1,b^o) \leq 0 \\ &\implies \sum_{o\in\mathcal{O}} f_O(o|1)V^*_{\pi_1,l}(1,b^o) \leq \sum_{o\in\mathcal{O}} f_O(o|0)V^*_{\pi_1,l}(0,b^o) \\ &\implies \pi_2^*(0,b(1)) = C\end{aligned} \tag{37}$$

□

Lemma 6. *If $\pi_1(S|b(1))$ is non-decreasing with $b(1)$ and f_O is TP2, then $V^*_{\pi_1,l}\big(0,b(1)\big) - V^*_{\pi_1,l}\big(1,b(1)\big)$ is non-decreasing with $\pi_1(S|b(1))$.*

Proof. We prove this statement by mathematical induction. Let $W^k_{\pi_1,l}(b(1)) = V^k_{\pi_1,l}\big(0,b(1)\big) - V^k_{\pi_1,l}\big(1,b(1)\big)$. For $k=1$, it follows from (7)-(12) that $W^1_{\pi_1,l}(b(1))$ is non-decreasing with $\pi_1(S|b(1)) \in [0,1]$. Assume by induction that the statement of Lemma 6 holds for $W^{k-1}_{\pi_1,l}(b(1))$. We show that then the statement holds also for $W^k_{\pi_1,l}(b(1))$.

There are three cases to consider:

- If $b(1) \in \mathscr{S}^{k,(2)}_{0,l,\pi_1} \cap \mathscr{C}^{k,(2)}_{1,l,\pi_1}$, then:

$$W^k_{\pi_1,l}(b(1)) = R_{int} + \pi_1(S|b(1))(R_{st}/l - R_{cost}/l - R_{int}) \tag{38}$$

  The right-hand side of (38) is non-decreasing with $\pi_1(S|b(1))$ since $R_{st}/l - R_{cost}/l - R_{int} \geq 0$ (see Section III).

- If $b(1) \in \mathscr{C}^{k,(2)}_{0,l,\pi_1} \cap \mathscr{C}^{k,(2)}_{1,l,\pi_1}$, then:

$$\begin{aligned}W^k_{\pi_1,l}(b(1)) = \pi_1(S|b(1))\Big(&R_{st}/l - R_{cost}/l - R_{int}\Big) + V^{k-1}_{\pi_1,l}\big(1,b(1)\big)\Big) + R_{int} + \sum_o f_O(o|0) \\ &V^k_{\pi_1,l}\big(0,b^o(1)\big) - (1-\phi_l)f_O(o|1)V^k_{\pi_1,l}\big(1,b^o(1)\big)\end{aligned} \tag{39}$$

  The first term in the right-hand side of (39) is non-decreasing with $\pi_1(S|b(1))$ since $R_{st}/l - R_{cost}/l - R_{int} \geq 0$ (see Section III) and $V^{k-1}_{\pi_1,l}\big(1,b(1)\big) \geq 0$ (it is

a consequence of Lemma 1 and (21)-(23)). The second term is non-decreasing with $\pi_1(S|b(1))$ by the induction hypothesis and the assumption that f_O is TP2.

- If $b(1) \in \mathscr{C}^{k,(2)}_{0,l,\pi_1} \cap \mathscr{S}^{k,(2)}_{1,l,\pi_1}$, then:

$$W^k_{\pi_1,l}(b(1)) = \pi_1(S|b(1))(-R_{cost}/l) \tag{40}$$
$$+ \sum_o f_O(o|0) V^k_{\pi_1,l}\big(0, b^o(1)\big)$$
$$= \pi_1(S|b(1))(-R_{cost}/l) + \sum_o f_O(o|0) \cdot \tag{41}$$
$$V^k_{\pi_1,l}\big(0, b^o(1)\big) - (1-\phi_l) f_O(o|1) V^k_{\pi_1,l}\big(1, b^o(1)\big)$$

The first term in the right-hand side of (40) is non-decreasing with $\pi_1(S|b(1))$ since $-R_{cost}/l \geq 0$. The second term is non-decreasing with $\pi_1(S|b(1))$ by the induction hypothesis and the assumption that f_O is TP2. (41) follows from Lemma 3 and the fact that $b(1) \in \mathscr{S}^{k,(2)}_{1,l,\pi_1}$.

The other cases, e.g. $b(1) \in \mathscr{S}^{k,(2)}_{0,l,\pi_1} \cap \mathscr{S}^{k,(2)}_{1,l,\pi_1}$, can be discarded due to Lemma 5. Hence, $W^k_{\pi_1,l}(b(1))$ is non-decreasing with $\pi_1(S|b(1))$ 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: $\mathscr{S}^{(2)}_{1,l,\pi_1} = [\tilde{\beta}_{1,l}, 1]$, and that the continuation sets in state $s = 0$ have the form: $\mathscr{C}^{(2)}_{0,l,\pi_1} = [\tilde{\beta}_{0,l}, 1]$, for some values $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$.

Proof of Theorem 1.C. We first show that $1 \in \mathscr{S}^{(2)}_{1,l,\pi_1}$ and that $1 \in \mathscr{C}^{(2)}_{0,l,\pi_1}$. Since $\pi_1(S|1) = 1$, it follows from Lemma 4 that $1 \in \mathscr{S}^{(2)}_{1,l,\pi_1}$ and as a consequence of (21)-(23) we have that $\tilde{\pi}_2(0, b(1)) = C$ iff:

$$\sum_o f_O(o|0) V^*_{\pi_1,l-1}(0, b^o(1)) -$$
$$f_O(o|1) V^*_{\pi_1,l-1}(1, b^o(1)) \geq 0 \tag{42}$$

The left-hand side of the above equation is positive since a) f_O is assumed to be TP2; b) $\sum_o f_O(o|0) V^*_{\pi_1,l-1}(0, b^o(1)) \geq 0$ (Lemma 1); and c) $f_O(o|1) V^*_{\pi_1,l-1}(1, b^o(1)) = 0$ (Lemma 3). Hence, $1 \in \mathscr{C}^{(2)}_{0,l,\pi_1}$.

Now we show that $\mathscr{S}^{(2)}_{1,l,\pi_1} = [\tilde{\beta}_{1,l}, 1]$ and that $\mathscr{C}^{(2)}_{0,l,\pi_1} = [\tilde{\beta}_{0,l}, 1]$ for some values $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$. From (21)-(23) we know that $\tilde{\pi}_2(1, b(1)) = S$ iff:

$$\mathbb{E}_{\pi_1}\Big[\mathcal{R}\big(1, (a^{(1)}, C)\big) + \tag{43}$$
$$(\phi_l - 1) \sum_o f_O(o|1) V^*_{\pi_1,l-a^{(1)}}(1, b^o(1))\Big] \geq 0$$

The first term in the left-hand side of (43) is increasing with $b(1)$ (7)-(12). Further, it follows from Lemma 2 that the second term is decreasing with $b(1)$. Hence, we conclude that if $\tilde{\pi}_2(1, b(1)) = S$, then for any $b'(1) \geq b(1)$, $\tilde{\pi}_2(1, b'(1)) = S$. As a consequence, there exist values $\tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{1,L}$ such that $\mathscr{S}^{(2)}_{1,l,\pi_1} = [\tilde{\beta}_{1,l}, 1]$.

Similarly, from (21)-(23) we know that $\tilde{\pi}_2(0, b(1)) = C$ iff:

$$\mathbb{E}_{\pi_1}\Big[\sum_o f_O(o|0) V^*_{\pi_1,l-a^{(1)}}(0, b^o(1)) \tag{44}$$
$$- f_O(o|1) V^*_{\pi_1,l-a^{(1)}}(1, b^o(1))\Big] \geq 0$$

Since f_O is TP2 and $\pi_1(S|b(1))$ is increasing with $b(1)$, the left-hand side in (44) is decreasing (it follows from Lemma 2 and Lemma 6). Hence, we conclude that if $\tilde{\pi}_2(0, b(1)) = C$, then for any $b'(1) \geq b(1)$, $\tilde{\pi}_2(0, b'(1)) = C$. As a result, there exist values $\tilde{\beta}_{0,1}, \ldots, \tilde{\beta}_{0,L}$ such that $\mathscr{C}^{(2)}_{0,l,\pi_1} = [\tilde{\beta}_{0,l}, 1]$. □

Game Parameters	**Values**
$R_{st}, R_{cost}, R_{int}, \gamma, \phi_l, L$	20, -2, -1, 0.99, $1/2l$, 7
T-FP Parameters	**Values**
$c, \epsilon, \lambda, A, a, N, \delta$	10, 0.101, 0.602, 100, 1, 50, 0.2
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).

Appendix B
Hyperparameters

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

Appendix C
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] S. Ayoubi, N. Limam, M. A. Salahuddin, N. Shahriar, R. Boutaba, F. Estrada-Solano, and O. M. Caicedo, "Machine learning for cognitive network management," *IEEE Communications Magazine*, vol. 56, no. 1, pp. 158–165, 2018.

[3] M. Rasouli, E. Miehling, and D. Teneketzis, "A supervisory control approach to dynamic cyber-security," in *Decision and Game Theory for Security*, 2014.

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

[5] 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.

[6] U.-M. O'Reilly and E. Hemberg, "An artificial coevolutionary framework for adversarial ai," in *AAAI Fall Symposium: ALEC*, 2018.

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

[8] 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.

[9] K. Hammar and R. Stadler, "Intrusion prevention through optimal stopping," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2333–2348, 2022.

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

[11] ——, "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.

[12] 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.

[13] 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.

[14] 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.

[15] T. T. Nguyen and V. J. Reddi, "Deep reinforcement learning for cyber security," *IEEE transactions on neural networks and learning systems*, vol. PP, 2021.

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

[17] K. Hammar and R. Stadler, "An online framework for adapting security policies in dynamic it environments," in *2022 18th International Conference on Network and Service Management (CNSM)*, 2022, pp. 359–363.

[18] 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.

[19] 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, Jun. 2020.

[20] 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.

[21] 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.

[22] W. Blum, "Gamifying machine learning for stronger security and ai models," 2021, microsoft Research.

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

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] 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.

[29] 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.

[30] 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.

[31] 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.

[32] K. Hammar and R. Stadler, "A system for interactive examination of learned security policies," in *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, 2022, pp. 1–3.

[33] L. Zhang, Y. Pan, Y. Liu, Q. Zheng, and Z. Pan, "Multiple domain cyberspace attack and defense game based on reward randomization reinforcement learning," 2022.

[34] A. Dutta, E. Al-Shaer, and S. Chatterjee, "Constraints satisfiability driven reinforcement learning for autonomous cyber defense," *CoRR*, vol. abs/2104.08994, 2021.

[35] Y. Du, Z. Song, S. Milani, C. Gonzales, and F. Fang, "Learning to play an adaptive cyber deception game," *The 13th Workshop on Optimization and Learning in Multiagent Systems, AAMAS 2022*, 2022.

[36] N. M. Yungaicela-Naula, C. Vargas-Rosales, J. A. Pérez-Díaz, and D. F. Carrera, "A flexible sdn-based framework for slow-rate ddos attack mitigation by using deep reinforcement learning," *Journal of Network and Computer Applications*, vol. 205, p. 103444, 2022.

[37] M. Zolotukhin, S. Kumar, and T. Hämäläinen, "Reinforcement learning for attack mitigation in sdn-enabled networks," in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, 2020, pp. 282–286.

[38] T. Zhu, D. Ye, Z. Cheng, W. Zhou, and P. S. Yu, "Learning games for defending advanced persistent threats in cyber systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2022.

[39] Y. Liu, K.-F. Tsang, C. K. Wu, Y. Wei, H. Wang, and H. Zhu, "Ieee p2668-compliant multi-layer iot-ddos defense system using deep reinforcement learning," *IEEE Transactions on Consumer Electronics*, pp. 1–1, 2022.

[40] R. R. dos Santos, E. K. Viegas, A. O. Santin, and V. V. Cogo, "Reinforcement learning for intrusion detection: More model longness and fewer updates," *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022.

[41] H. Liu, Y. Li, J. Mårtensson, L. Xie, and K. H. Johansson, "Reinforcement learning based approach for flip attack detection," in *2020 59th IEEE Conference on Decision and Control (CDC)*, 2020.

[42] R. Maeda and M. Mimura, "Automating post-exploitation with deep reinforcement learning," *Computers & Security*, vol. 100, 2021.

[43] J. A. Bland, M. D. Petty, T. S. Whitaker, K. P. Maxwell, and W. A. Cantrell, "Machine learning cyberattack and defense strategies," *Computers & Security*, vol. 92, p. 101738, 2020.

[44] L. Huang and Q. Zhu, "Radams: Resilient and adaptive alert and attention management strategy against informational denial-of-service (ids) attacks," *Computers & Security*, vol. 121, p. 102844, 2022.

[45] X. Liu, H. Zhang, S. Dong, and Y. Zhang, "Network defense decision-making based on a stochastic game system and a deep recurrent q-network," *Computers & Security*, vol. 111, p. 102480, 2021.

[46] P. Zhang, C. Wang, C. Jiang, and A. Benslimane, "Security-aware virtual network embedding algorithm based on reinforcement learning," *IEEE Transactions on Network Science and Engineering*, vol. 8, no. 2, pp. 1095–1105, 2021.

[47] J. Khoury and M. E. B. Nassar, "A hybrid game theory and reinforcement learning approach for cyber-physical systems security," *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pp. 1–9, 2020.

[48] S. Wang, Q. Pei, J. Wang, G. Tang, Y. Zhang, and X. Liu, "An intelligent deployment policy for deception resources based on reinforcement learning," *IEEE Access*, vol. 8, pp. 35 792–35 804, 2020.

[49] Y. Han *et al.*, "Reinforcement learning for autonomous defence in software-defined networking," in *Decision and Game Theory for Security*, 2018, pp. 145–165.

[50] Y. Guo, Z. Wu, L. Tian, Y. Wang, J. Xie, Y. Du, and Y. Zhang, "Network security defense decision-making method based on stochastic game and deep reinforcement learning," *Security and Communication Networks*, vol. 2021, p. 2283786, 2021.

[51] S. Iannucci, E. Casalicchio, and M. Lucantonio, "An intrusion response approach for elastic applications based on reinforcement learning," *2021 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 01–10, 2021.

[52] S. Iannucci, O. D. Barba, V. Cardellini, and I. Banicescu, "A performance evaluation of deep reinforcement learning for model-based intrusion response," *2019 IEEE 4th International Workshops on Foundations and Applications of Self* Systems (FAS*W)*, pp. 158–163, 2019.

[53] M. Wolk, A. Applebaum, C. Dennler, P. Dwyer, M. Moskowitz, H. Nguyen, N. Nichols, N. Park, P. Rachwalski, F. Rau, and A. Webster, "Beyond cage: Investigating generalization of learned autonomous network defense policies," 2022.

[54] L. Zhang, T. Zhu, F. K. Hussain, D. Ye, and W. Zhou, "Defend to defeat: Limiting information leakage in defending against advanced persistent threats," *IEEE Transactions on Information Forensics and Security*, pp. 1–1, 2022.

[55] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester, "Challenges of real-world reinforcement learning: definitions, benchmarks and analysis," *Machine Learning*, vol. 110, no. 9, pp. 2419–2468, 2021.

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

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

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

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

[60] S. Iannucci, Q. Chen, and S. Abdelwahed, "High-performance intrusion response planning on many-core architectures," in *2016 25th International Conference on Computer Communication and Networks (ICCCN)*, 2016, pp. 1–6.

[61] O. P. Kreidl and T. M. Frazier, "Feedback control applied to survivability: a host-based autonomic defense system," *IEEE Transactions on Reliability*, vol. 53, pp. 148–166, 2004.

[62] S. Iannucci and S. Abdelwahed, "A probabilistic approach to autonomic security management," in *2016 IEEE International Conference on Autonomic Computing (ICAC)*, 2016, pp. 157–166.

[63] 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.

[64] A. Applebaum, C. Dennler, P. Dwyer, M. Moskowitz, H. Nguyen, N. Nichols, N. Park, P. Rachwalski, F. Rau, A. Webster, and M. Wolk, "Bridging automated to autonomous cyber defense: Foundational analysis of tabular q-learning," in *Proceedings of the 15th ACM Workshop on Artificial Intelligence and Security*, ser. AISec'22. New York, NY, USA: Association for Computing Machinery, 2022, p. 149–159.

[65] Y. Shi, Y. E. Sagduyu, T. Erpek, and M. C. Gursoy, "How to attack and defend nextg radio access network slicing with reinforcement learning," *IEEE Open Journal of Vehicular Technology*, pp. 1–11, 2022.

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

[67] 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.

[68] 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.

[69] Q. Xu, Z. Su, and R. Lu, "Game theory and reinforcement learning based secure edge caching in mobile social networks," *IEEE Transactions on Information Forensics and Security*, vol. 15, 2020.

[70] K. Li, B. Jiu, W. Pu, H. Liu, and X. Peng, "Neural fictitious self-play for radar anti-jamming dynamic game with imperfect information," *IEEE Transactions on Aerospace and Electronic Systems*, pp. 1–1, 2022.

[71] 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, 2020.

[72] B. Wang, Y. L. Sun, M. Sun, and X. Xu, "Game-theoretic actor–critic-based intrusion response scheme (gtac-irs) for wireless sdn-based iot networks," *IEEE Internet of Things Journal*, vol. 8, pp. 1830–1845, 2021.

[73] Trellix, "Trellix intrusion prevention system," 2022.

[74] W. Inc, "Wazuh - the open source security platform," 2022.

[75] 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.

[76] K. Hammar and R. Stadler, "Learning security strategies through game play and optimal stopping," in *Proceedings of the ML4Cyber workshop, ICML 2022, Baltimore, USA, July 17-23, 2022*. PMLR, 2022.

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

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

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

[80] 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.

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

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

[83] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed., USA, 1994.

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

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

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

[87] 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.

[88] 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.

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

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

[91] 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.

[92] 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.

[93] D. Merkel, "Docker: lightweight linux containers for consistent development and deployment," *Linux journal*, vol. 2014, p. 2, 2014.

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

[95] 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.

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

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

[98] T. M. Corporation, "Cve database," 2022, https://cve.mitre.org/.

[99] 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.

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

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

[102] 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.

[103] W. Xue, Y. Zhang, S. Li, X. Wang, B. An, and C. K. Yeo, "Solving large-scale extensive-form network security games via neural fictitious self-play," 2021.

[104] 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.

[105] P. Tomášek, B. Bosansky, and T. Nguyen, *Using One-Sided Partially Observable Stochastic Games for Solving Zero-Sum Security Games with Sequential Attacks*, 12 2020, pp. 385–404.

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

[107] T. H. Nguyen, M. Wright, M. P. Wellman, and S. Singh, "Multistage attack graph security games: Heuristic strategies, with empirical game-theoretic analysis," *Security and Communication Networks*, vol. 2018, p. 2864873, Dec 2018.

[108] C. Bakker, A. Bhattacharya, S. Chatterjee, and D. L. Vrabie, "Learning and information manipulation: Repeated hypergames for cyber-physical security," *IEEE Control Systems Letters*, vol. 4, no. 2, 2020.

[109] Z. Wan, J.-H. Cho, M. Zhu, A. H. Anwar, C. A. Kamhoua, and M. P. Singh, "Foureye: Defensive deception against advanced persistent threats via hypergame theory," *IEEE Transactions on Network and Service Management*, vol. 19, no. 1, pp. 112–129, 2022.

[110] H. Zhang, J. Tan, X. Liu, S. Huang, H. Hu, and Y. Zhang, "Cybersecurity threat assessment integrating qualitative differential and evolutionary games," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 3425–3437, 2022.

[111] H. Hu, Y. Liu, C. Chen, H. Zhang, and Y. Liu, "Optimal decision making approach for cyber security defense using evolutionary game," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1683–1700, 2020.

[112] 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.

[113] 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.

[114] 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.

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

[116] 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.

[117] 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.

[118] 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.

[119] 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.

[120] 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.

[121] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley, "Rre: A game-theoretic intrusion response and recovery engine," in *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, 2009, pp. 439–448.

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

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

[124] 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.

[125] 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.

[126] 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.

[127] P. Holgado, V. A. Villagrá, and L. Vázquez, "Real-time multistep attack prediction based on hidden markov models," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 1, pp. 134–147, 2020.

[128] 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.

[129] I. Siniosoglou, P. Radoglou-Grammatikis, G. Efstathopoulos, P. Fouliras, and P. Sarigiannidis, "A unified deep learning anomaly detection and classification approach for smart grid environments," *IEEE Transactions on Network and Service Management*, vol. 18, no. 2, 2021.

[130] D. Anderson, T. Frivold, and A. Valdes, "Next-generation intrusion detection expert system (nides) a summary," 01 1995.

[131] R. Heenan and N. Moradpoor, "Introduction to security onion," in *The First Post Graduate Cyber Security Symposium*, 2016.

[132] V. Paxson, "Bro: a system for detecting network intruders in real-time," *Computer networks*, vol. 31, no. 23-24, pp. 2435–2463, 1999.

[133] S. Lewandowski, D. Van Hook, G. O'Leary, J. Haines, and L. Rossey, "Sara: Survivable autonomic response architecture," in *Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX'01*, vol. 1, 2001, pp. 77–88 vol.1.

[134] B. Foo, Y.-C. Mao, and E. Spafford, "Adepts: Adaptive intrusion response using attack graphs in an e-commerce environment," in *Proceedings of the 2005 International Conference on Dependable Systems and Networks*, ser. DSN '05, USA, 2005, p. 508–517.

[135] A. Uprety and D. B. Rawat, "Reinforcement learning for iot security: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 8, no. 11, pp. 8693–8706, 2021.

[136] S. "Jajodia, G. Cybenko, P. Liu, C. Wang, and M. Wellman, *Adversarial and Uncertain Reasoning for Adaptive Cyber Defense: Control- and Game-Theoretic Approaches to Cyber Security*. Cham: Springer International Publishing, 2019.

[137] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tilbury, *Feedback Control of Computing Systems*. USA: Wiley & Sons, 2004.

[138] A. Andrew, S. Spillard, J. Collyer, and N. Dhir, "Developing optimal causal cyber-defence agents via cyber security simulation," in *Proceedings of the ML4Cyber workshop, ICML 2022, Baltimore, USA, July 17-23, 2022*. PMLR, 2022.

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

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

[141] 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.

[142] J. Collyer, A. Andrew, and D. Hodges, "Acd-g: Enhancing autonomous cyber defense agent generalization through graph embedded network representation," in *Proceedings of the ML4Cyber workshop, ICML 2022, Baltimore, USA, July 17-23, 2022*. PMLR, 2022.

[143] S. G. Aksoy, E. Purvine, and S. J. Young, "Directional laplacian centrality for cyber situational awareness," *Digital Threats*, oct 2021.

[144] J. Wang, C. Song, and H. Yin, "Reinforcement learning-based hierarchical seed scheduling for greybox fuzzing," in *NDSS*, 2021.

[145] T. Avgerinos, D. Brumley, J. Davis, R. Goulden, T. Nighswander, A. Rebert, and N. Williamson, "The mayhem cyber reasoning system," *IEEE Security Privacy*, vol. 16, no. 2, pp. 52–60, March 2018.

[146] A. N. Kulkarni and J. Fu, "A theory of hypergames on graphs for synthesizing dynamic cyber defense with deception," 2020.

[147] 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.

[148] J. A. Emanuello and A. Ridley, "The mathematics of cyber defense," 2022, notices of the American Mathematical Society (AMS).

[149] E. Al-Shaer, J. Wei, K. W. Hamlen, and C. Wang, *Autonomous Cyber Deception - Reasoning, Adaptive Planning, and Evaluation of HoneyThings*. Springer, 2019.

[150] K. Han, J. H. Choi, Y. Choi, G. M. Lee, and A. B. Whinston, "Security defense against long-term and stealthy cyberattacks," *Decision Support Systems*, p. 113912, 2022.

[151] 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.

[152] J. Brynielsson and S. Arnborg, "Bayesian games for threat prediction and situation analysis," in *Proceedings of the 7th International Conference on Information Fusion*, vol. 2, Stockholm, Jun. 2004.

[153] U. Franke and J. Brynielsson, "Cyber situational awareness - a systematic review of the literature," *Comput. Secur.*, vol. 46, 2014.

[154] S. Sengupta and S. Kambhampati, "Multi-agent reinforcement learning in bayesian stackelberg markov games for adaptive moving target defense," *CoRR*, vol. abs/2007.10457, 2020.

[155] 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.

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

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

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

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

[160] 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.

[161] 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: Association for Computing Machinery, 2010.
[162] A. Teixeira, S. Amin, H. Sandberg, K. H. Johansson, and S. S. Sastry, "Cyber security analysis of state estimators in electric power systems," in *49th IEEE Conference on Decision and Control (CDC)*, 2010.
[163] A. Teixeira, K. C. Sou, H. Sandberg, and K. H. Johansson, "Secure control systems: A quantitative risk management approach," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 24–45, 2015.
[164] H. Sandberg, S. Amin, and K. H. Johansson, "Cyberphysical security in networked control systems: An introduction to the issue," *IEEE Control Systems Magazine*, vol. 35, no. 1, pp. 20–23, 2015.
[165] M. S. Chong, H. Sandberg, and A. M. Teixeira, "A tutorial introduction to security and privacy for cyber-physical systems," in *2019 18th European Control Conference (ECC)*, 2019, pp. 968–978.
[166] A. Leva, M. Maggio, A. V. Papadopoulos, and F. Terraneo, *Control-Based Operating System Design*. Institution of Engineering and Technology, 2013.
[167] M. Rasouli, E. Miehling, and D. Teneketzis, "A scalable decomposition method for the dynamic defense of cyber networks," in *Game Theory for Security and Risk Management: From Theory to Practice*, 2018.
[168] ——, "A supervisory control approach to dynamic cyber-security," in *Decision and Game Theory for Security*, 2014.
[169] E. Miehling, M. Rasouli, and D. Teneketzis, "Optimal defense policies for partially observable spreading processes on bayesian attack graphs," in *Proceedings of the 2nd ACM Workshop on Moving Target Defense*, ser. MTD '15, New York, 2015, p. 67–76.
[170] R. Bellman, *Dynamic Programming*. Dover Publications, 1957.
[171] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-dynamic programming*. Belmont, MA: Athena Scientific, 1996.
[172] D. Silver *et al.*, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.
[173] C. Berner *et al.*, "Dota 2 with large scale deep reinforcement learning," *ArXiv*, vol. abs/1912.06680, 2019.
[174] 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.
[175] L. Li, R. Fayad, and A. Taylor, "Cygil: A cyber gym for training autonomous agents over emulated network systems," *CoRR*, vol. abs/2109.03331, 2021.
[176] A. Molina-Markham, C. Miniter, B. Powell, and A. Ridley, "Network environment design for autonomous cyberdefense," 2021, https://arxiv.org/abs/2103.07583.
[177] D. Bertsekas, *Rollout, Policy Iteration, and Distributed Reinforcement Learning*, ser. Athena scientific optimization and computation series. Athena Scientific., 2021.