Learning Security Strategies through Game Play and Optimal Stopping

Kim Hammar [1 2 3] **Rolf Stadler** [1 2 3]

Abstract

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

1. Introduction

We present a novel approach to automatically learn security strategies for an attacker and a defender. We apply this approach to an *intrusion prevention* use case, which involves the IT infrastructure of an organization (see Fig. 1). The operator of this infrastructure, which we call the defender, takes measures to protect it against a possible attacker while providing services to a client population. (We use the term

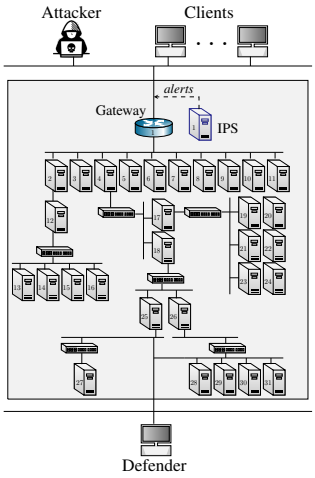

Figure 1. The IT infrastructure and the actors in the use case.

"intrusion prevention" as suggested in the literature, e.g. in (Fuchsberger, 2005). It means that the attacker is prevented from reaching its goal, rather than prevented from accessing any part of the infrastructure.)

We formulate the use case as an *optimal stopping game*, i.e. a stochastic game where each player faces an optimal stopping problem (Wald, 1947; Dynkin, 1969). The stopping game formulation enables us to gain insight into the structure of optimal strategies, which we show to have threshold properties. To obtain effective defender strategies, we use reinforcement learning and self-play. Based on the threshold properties of optimal strategies, we design an efficient self-play algorithm that iteratively computes optimal defender strategies against a dynamic attacker.

Our method for learning and evaluating strategies includes building two systems (see Fig. 2). First, we develop an *emulation system* where key functional components of the target infrastructure are replicated. This system closely approximates the functionality of the target infrastructure and is used to run attack scenarios and defender responses. These runs produce system statistics and logs, from which we estimate distributions of infrastructure metrics. We then use the estimated distributions to instantiate the simulation model. Second, we build a *simulation system* where game episodes are simulated and strategies are incremen-

[1]Division of Network and Systems Engineering [2]KTH Center for Cyber Defense and Information Security [3]KTH Royal Institute of Technology, Stockholm, Sweden. Correspondence to: Kim Hammar <kimham@kth.se>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

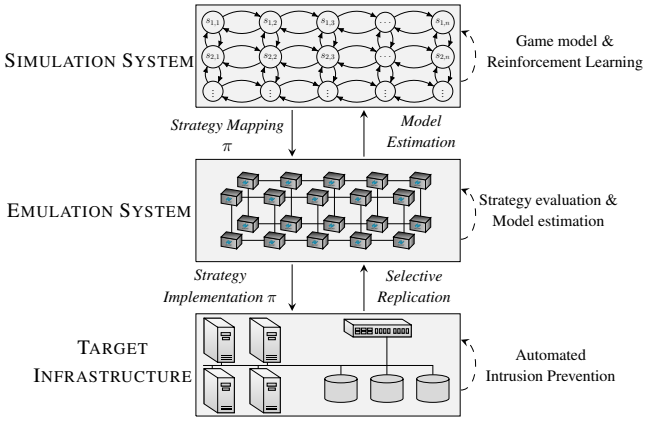

Figure 2. Our approach for finding and evaluating intrusion prevention strategies.

tally learned. Learned strategies are then extracted from the simulation system and evaluated in the emulation system. (A video demonstration of our software framework that implements the emulation and simulation systems is available at (Hammar & Stadler, 2022a).)

We make three contributions with this paper. First, we formulate intrusion prevention as an optimal stopping game. This novel formulation allows us a) to derive structural properties of optimal strategies using results from optimal stopping theory; and b) to find defender strategies that are effective against attackers with dynamic strategies. We thus address a limitation of many related works that consider static attackers only (Ridley, 2018; Blum, 2021; Tran et al., 2021; Hammar & Stadler, 2021; 2022b; Akbari et al., 2020; Liu et al., 2018). Second, we propose T-FP, an efficient reinforcement learning algorithm that exploits structural properties of optimal stopping strategies and outperforms a state-of-the-art algorithm for our use case. Third, we provide evaluation results from an emulated infrastructure. This addresses a common drawback in related research that relies on simulations to learn and evaluate strategies (Blum, 2021; Ridley, 2018; Tran et al., 2021; Schwartz et al., 2020; Hammar & Stadler, 2020).

2. The Intrusion Prevention Use Case

We consider an intrusion prevention use case that involves the IT infrastructure of an organization. The operator of this infrastructure, which we call the defender, takes measures to protect it against an attacker while providing services to a client population (Fig. 1). The infrastructure includes a set of servers that run the services and an Intrusion Prevention System (IPS) that logs events in real-time. Clients access the services through a public gateway, which also is 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 getting detected. 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 IPS alerts and other statistics. It can take a fixed number of defensive actions, each of which has a cost and a chance of preventing an ongoing attack. An example of a defensive action is to drop network traffic that triggers IPS 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 ongoing 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*), prevent a possible intrusion at 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 prevented at minimal cost through defensive actions. The challenge for the defender is to identify the precise time for this moment.

3. Formalizing The Use Case

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

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

capabilities.

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

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

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

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

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

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

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

Both players have perfect recall, meaning that they remember their respective play history. The history of the defender at time-step t is $h_t^{(1)} = (\rho_1, a_1^{(1)}, o_1, \ldots, a_{t-1}^{(1)}, o_t)$ 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 s_t, which is expressed in the *belief state* $b_t(s_t) = \mathbb{P}[s_t | h_t^{(1)}] \in \mathcal{B}$. Since $s_t \in \{0, 1\}$ and $b_t(0) = 1 - b_t(1)$, b_t is determined by $b_t(1)$. Hence, we can model $\mathcal{B} = [0, 1]$.

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

$$\mathcal{T}_{l_t>1}\big(0, 0, (S, C)\big) = \mathcal{T}_{l_t}\big(0, 0, (C, C)\big) = 1 \tag{1}$$
$$\mathcal{T}_{l_t>1}\big(1, 1, (\cdot, C)\big) = \mathcal{T}_{l_t}\big(1, 1, (C, C)\big) = 1 - \phi_{l_t} \tag{2}$$
$$\mathcal{T}_{l_t>1}\big(1, 0, (\cdot, S)\big) = \mathcal{T}_{l_t}\big(1, 0, (C, S)\big) = 1 \tag{3}$$
$$\mathcal{T}_{l_t>1}\big(\emptyset, 1, (\cdot, C)\big) = \mathcal{T}_{l_t}\big(\emptyset, 1, (C, C)\big) = \phi_{l_t} \tag{4}$$
$$\mathcal{T}_1\big(\emptyset, \cdot, (S, \cdot)\big) = \mathcal{T}_{l_t}(\emptyset, \emptyset, \cdot) = \mathcal{T}_{l_t}(\emptyset, 1, (\cdot, S)) = 1 \tag{5}$$

All other state transitions have probability 0.

Eqs. 1-2 define the probabilities of the recurrent state transitions $0 \to 0$ 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 $1 - \phi_{l_t}$ if the attacker chooses action C and $l_t - a_t^{(1)} > 0$. ϕ_{l_t} is a parameter of the use case that defines the probability that the intrusion is prevented, which increases with each stop action that the defender takes.

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

A game *episode* starts at $t = 1$ and ends at $t = T$. The time horizon T is a random variable that depends on both players' strategies and takes values in $\{2, 3, \ldots, \infty\}$.

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

$$\mathcal{R}_{l_t}(\emptyset, \cdot) = 0, \qquad \mathcal{R}_{l_t}\big(1, (\cdot, S)\big) = 0 \tag{6}$$
$$\mathcal{R}_{l_t}\big(0, (C, \cdot)\big) = 0 \tag{7}$$
$$\mathcal{R}_{l_t}\big(0, (S, \cdot)\big) = R_{cost}/l_t \tag{8}$$
$$\mathcal{R}_{l_t}\big(1, (S, C)\big) = R_{st}/l_t \tag{9}$$
$$\mathcal{R}_{l_t}\big(1, (C, C)\big) = R_{int} \tag{10}$$

Eq. 6 states that the reward is zero in the terminal state and when the attacker ends an intrusion. Eq. 7 states that the defender incurs no cost when it is not under attack and not taking defensive actions. Eq. 8 indicates that the defender incurs a cost when stopping if no intrusion is ongoing, which is decreasing with the number of stops remaining l_t. Eq. 9

states that the defender receives a reward that is decreasing in l_t when taking a stop action that affects an ongoing intrusion. Lastly, Eq. 10 indicates that the defender receives a loss for each time-step when under intrusion.

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 $\mathcal{Z}(o_t, s_t, (a^{(1)}_{t-1}, a^{(2)}_{t-1}))= \mathbb{P}[o_t|s_t, (a^{(1)}_{t-1}, a^{(2)}_{t-1})]$ as follows:

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

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

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

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

$$b_{t+1}(s_{t+1}) = C \sum_{s_t \in \mathcal{S}} \sum_{a^{(2)}_t \in \mathcal{A}_2} \sum_{o_{t+1} \in \mathcal{O}} b_t(s_t)\pi_{2,l}(a^{(2)}_t|s_t, b_t) \cdot \mathcal{Z}(o_{t+1}, s_{t+1}, (a^{(1)}_t, a^{(2)}_t))\mathcal{T}\big(s_{t+1}, s_t, (a^{(1)}_t, a^{(2)}_t)\big) \tag{14}$$

where $C = 1/\mathbb{P}[o_{t+1}|a^{(1)}_1, \pi_{2,l}, b_t]$ is a normalizing factor that makes b_{t+1} sum to 1. The initial belief is $b_1(0) = 1$.

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

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

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

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

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

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

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

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

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

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

This means that no player has an incentive to change its strategy and that $(\pi^*_{1,l}, \pi^*_{2,l})$ is a Nash equilibrium (1951).

4. Game-Theoretic Analysis and Finding Optimal Defender Strategies

Finding optimal strategies that satisfy Eq. 19 means finding a Nash equilibrium for the POSG Γ. We know from game theory that Γ has at least one mixed Nash equilibrium (von Neumann, 1928; Nash, 1951; Horák, 2019). (A Nash equilibrium is called mixed if one or more players follow mixed strategies.)

The equilibria of Γ can be obtained by finding pairs of strategies that are best responses against each other (Eq. 19). A best response for the defender is obtained by solving a POMDP $\mathcal{M}^P$, and a best response for the attacker is obtained by solving an MDP $\mathcal{M}$. Since the game is zero-sum, stationary, and $\gamma < 1$, it follows from Markov decision theory that for any strategy pair $(\pi_{1,l}, \pi_{2,l})$, a corresponding pair of best response strategies $(\tilde{\pi}_{1,l} \in B_1(\pi_{2,l}), \tilde{\pi}_{2,l} \in B_2(\pi_{1,l}))$ exists (Puterman, 1994; Bellman, 1957).

Analyzing best responses using optimal stopping. The POMDP $\mathcal{M}^P$ and the MDP $\mathcal{M}$ that determine the best response strategies can be understood as *optimal stopping* problems (Wald, 1947; Dynkin, 1969).

In the defender's case, the problem is to find a stopping strategy $\pi^*_{1,l}(b_t) \to \{S, C\}$ that maximizes J_1 (Eq. 15) and prescribes the optimal stopping times $\tau^*_{1,1}, \tau^*_{1,2}, \ldots, \tau^*_{1,L}$. Similarly, the problem for the attacker is to find a stopping strategy $\pi^*_{2,l}(s_t, b_t) \to \{S, C\}$ that minimizes J_1 (Eq. 16) and prescribes the optimal stopping times $\tau^*_{2,1}$ and $\tau^*_{2,2}$.

Based on (Hammar & Stadler, 2022b; Horák, 2019), we formulate Theorem 4.1, which contains an existence result for equilibria and a structural result for best response strategies in the game.

Theorem 4.1. *Given the one-sided POSG Γ in Section 3 with $L \geq 1$, the following holds.*

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

(B) Given any attacker strategy $\pi_{2,l} \in \Pi_2$, if the probability mass function $f_{O|s}$ is totally positive of order 2

(i.e., TP2 (Hammar & Stadler, 2022b)), there exist values $\tilde{\alpha}_1 \geq \tilde{\alpha}_2 \geq \ldots \geq \tilde{\alpha}_L \in [0,1]$ *and a best response strategy* $\tilde{\pi}_{1,l} \in B_1(\pi_{2,l})$ *of the defender that satisfies:*

$$\tilde{\pi}_{1,l}(b(1)) = S \iff b(1) \geq \tilde{\alpha}_l, l \in 1, \ldots, L \quad (20)$$

(C) Given a defender strategy $\pi_{1,l} \in \Pi_1$*, where* $\pi_{1,l}(S|b(1))$ *is non-decreasing in* $b(1)$ *and* $\pi_{1,l}(S|1) = 1$*, there exist values* $\tilde{\beta}_{0,1}, \tilde{\beta}_{1,1}, \ldots, \tilde{\beta}_{0,L}, \tilde{\beta}_{1,L} \in [0,1]$ *and a best response strategy* $\tilde{\pi}_{2,l} \in B_2(\pi_{1,l})$ *of the attacker that satisfies:*

$$\tilde{\pi}_{2,l}(0, b(1)) = C \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{0,l} \quad (21)$$

$$\tilde{\pi}_{2,l}(1, b(1)) = S \iff \pi_{1,l}(S|b(1)) \geq \tilde{\beta}_{1,l} \quad (22)$$

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

Proof. The proof is available in the extended arXiv version of this paper (Hammar & Stadler, 2022c). Due to space limitation we do not include the proof here. □

Theorem 4.1 tells us that Γ has a mixed Nash equilibrium. It also tells us that, under certain assumptions, the best response strategies have threshold properties. In the following, we describe an efficient algorithm that takes advantage of these properties to approximate Nash equilibria of Γ.

Finding Nash equilibria through fictitious self-play. Computing Nash equilibria for a POSG is generally intractable (Horák, 2019). However, approximate solutions can be obtained through iterative approximation methods. One such method is *fictitious self-play*, where both players start from random strategies and continuously update their strategies based on the outcomes of played game episodes (Brown, 1951).

Fictitious self-play evolves through a sequence of iteration steps, which is illustrated in Fig. 3. An iteration step includes three procedures. 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, the iteration step is completed by having each player adopt 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 (Brown, 1951).

Our self-play algorithm: T-FP. We present a fictitious self-play algorithm, which we call T-FP, that exploits the statements in Theorem 4.1 to efficiently approximate Nash equilibria of Γ.

T-FP parameterizes the best response strategies $\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}} \in B_1(\pi_{2,l})$ and $\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}} \in B_2(\pi_{1,l})$ by threshold vectors. The

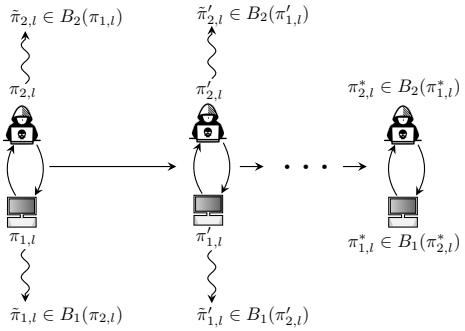

Figure 3. The fictitious self-play process; in every iteration each player learns a best response strategy $\tilde{\pi}_{i,l} \in B_i(\pi_{-i,l})$ and updates its strategy based on the empirical distribution of its past best responses; the horizontal arrows indicate the iterations of self-play and the vertical arrows indicate the learning of best responses; if the process is convergent, it reaches a Nash equilibrium $(\pi^*_{1,l}, \pi^*_{2,l})$.

defender's best response strategy is parameterized with the vector $\tilde{\theta}^{(1)} \in \mathbb{R}^L$ (Eq. 24). Similarly, the attacker's best response strategy is parameterized with the vector $\tilde{\theta}^{(2)} \in \mathbb{R}^{2L}$ (Eq. 25).

$$\varphi(a,b) = \left(1 + \left(\frac{b(1-\sigma(a))}{\sigma(a)(1-b)}\right)^{-20}\right)^{-1} \quad (23)$$

$$\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}\big(S|b(1)\big) = \varphi\left(\tilde{\theta}^{(1)}_l, b(1)\right) \quad (24)$$

$$\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}}\big(S|b(1), s\big) = \varphi\left(\tilde{\theta}^{(2)}_{sL+l}, \pi_{1,l}(S|b(1))\right) \quad (25)$$

$\sigma(\cdot)$ is the sigmoid function, $\sigma(\tilde{\theta}^{(1)}_1), \sigma(\tilde{\theta}^{(1)}_2), \ldots, \sigma(\tilde{\theta}^{(1)}_L) \in [0,1]$ are the L thresholds of the defender (see Theorem 4.1.B), and $\sigma(\tilde{\theta}^{(2)}_1), \sigma(\tilde{\theta}^{(2)}_2), \ldots, \sigma(\tilde{\theta}^{(2)}_{2L}) \in [0,1]$ are the $2L$ thresholds of the attacker (see Theorem 4.1.C).

Using this parameterization, T-FP learns best response strategies in each step of the self-play process by iteratively updating $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic approximation. To update the threshold vectors, T-FP simulates Γ, which allows to evaluate the objective functions $J_1(\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}, \pi_{2,l})$ (Eq. 15) and $J_2(\pi_{1,l}, \tilde{\pi}_{2,l,\tilde{\theta}^{(2)}})$ (Eq. 16). The obtained values of J_1 and J_2 are then used to estimate the gradients $\nabla_{\tilde{\theta}^{(1)}} J_1$ and $\nabla_{\tilde{\theta}^{(2)}} J_2$. Next, the estimated gradients are used to update $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ through stochastic gradient ascent. This procedure of estimating gradients and updating $\tilde{\theta}^{(1)}$ and $\tilde{\theta}^{(2)}$ continues until $\tilde{\pi}_{1,l,\tilde{\theta}^{(1)}}$ and $\tilde{\pi}_{2,l,\tilde{\theta}^{(2)}}$ have sufficiently converged.

The pseudocode of T-FP is listed in the extended arXiv version of this paper (Hammar & Stadler, 2022c). Due to space limitation we do not include it here.

5. Emulating the Target Infrastructure to Instantiate the Simulation

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

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. The system implements network isolation and traffic shaping on the containers using network namespaces and the NetEm module in the Linux kernel. Resource constraints on the containers, e.g. CPU and memory constraints, are enforced using cgroups.

The network topology of the emulated infrastructure is given in Fig. 1 and the configuration is given in (Hammar & Stadler, 2022c). The system emulates the clients, the attacker, the defender, network connectivity, and 31 physical components of the target infrastructure (e.g application servers and the gateway). The software functions replicate important components of the target infrastructure, such as, web servers, databases, and the Snort IPS, which is deployed using Snort's community ruleset v2.9.17.1.

We emulate connections between servers as full-duplex loss less connections with capacity 1 Gbit/s in both directions. We emulate external connections between the gateway and the client population as full-duplex connections of 100 Mbit/s capacity and 0.1% packet loss with random bursts of 1% packet loss. (These numbers are drawn from empirical studies on enterprise and wide area networks (Hammar & Stadler, 2022c).)

Emulating the client population. The *client population* is emulated by processes that run inside Docker containers and interact with the application servers through the gateway. The clients select functions uniformly at random from a fixed set, which is listed in (Hammar & Stadler, 2022c). We emulate client arrivals using a stationary Poisson process with parameter $\lambda = 20$ and exponentially distributed service times with parameter $\mu = \frac{1}{4}$. The duration of a time-step in the emulation is 30 seconds.

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

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

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

Table 1. Defender stop actions in the emulation.

Type	*Actions*
Reconnaissance	TCP-SYN scan, UDP port scan, TCP Null scan, TCP Xmas scan, TCP FIN scan, ping-scan, TCP connection scan, "Vulscan" 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 2. Attacker commands to emulate intrusions.

Like the defender, the attacker executes either a stop action or a continue action during each time-step. The attacker can take two stop actions. The first determines when the intrusion starts and the second determines when it ends (see Section 3). A continue action in state $s = 0$ has no affect on the emulation, but a continue action in state $s = 1$ has. When the attacker takes a stop action in state $s = 0$ or a continue action in state $s = 1$, an intrusion command is executed. We have implemented 25 such commands, which are listed in Table 2. During each step of an intrusion, the attacker selects a command uniformly at random from the list in Table 2.

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

Using these measurements, we fit a Gaussian mixture distribution $\hat{f}_O$ as an estimate of f_O in the target infrastructure

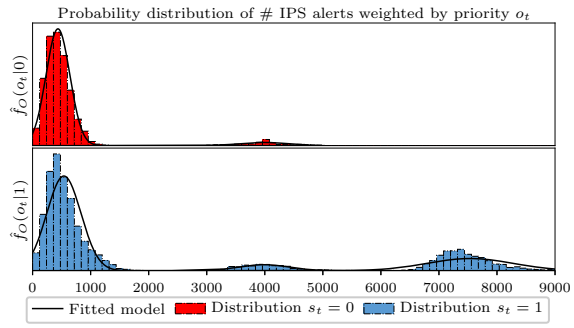

Figure 4. Empirical distributions of o_t when no intrusion occurs ($s_t = 0$) and during intrusion ($s_t = 1$); the black lines show the fitted Gaussian mixture models.

(Eqs. 11-12). For each state s, we obtain the conditional distribution $\hat{f}_{O|s}$ through expectation-maximization.

Fig. 4 shows the empirical distributions and the fitted model over the discrete observation space $\mathcal{O} = \{1, 2, \ldots, 9000\}$. $\hat{f}_{O|0}$ and $\hat{f}_{O|1}$ are Gaussian mixtures with two and three components, respectively. Both mixtures have most probability mass within the range $0 - 1000$. $\hat{f}_{O|1}$ also has substantial probability mass at larger values.

The stochastic matrix with the rows $\hat{f}_{O|0}$ and $\hat{f}_{O|1}$ has about 72×10^6 minors, out of which virtually all are non-negative. This suggests to us that the TP2 assumption in Theorem 4.1 can be made.

6. Learning Nash Equilibrium Strategies for the Target Infrastructure

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

We run T-FP for 500 iterations to estimate a Nash equilibrium using the iterative method described in Section 4. At the end of each iteration step, we evaluate the current strategy pair $(\pi_{1,l}, \pi_{2,l})$ by running 500 evaluation episodes in the simulation system and 5 evaluation episodes in the emulation system. This allows us to produce learning curves for different performance metrics (see Fig. 5). To estimate the convergence of the sequence of strategy pairs to a Nash equilibrium, we use the *exploitability* metric. The closer the exploitability is to zero, the closer the strategy pair is to a Nash equilibrium.

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

The hyperparameters, links to code, and the configurations of the environments for running simulations and emulations are available at (Hammar & Stadler, 2022c).

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

Baseline algorithms. We compare the performance of T-FP with two baseline algorithms: Neural Fictitious Self-Play (NFSP) (Heinrich & Silver, 2016) and Heuristic Search Value Iteration (HSVI) for one-sided POSGs (Horák et al., 2017). NFSP is a state-of-the-art deep reinforcement learning algorithm for imperfect-information games. Similar to T-FP, NFSP is a fictitious self-play algorithm. However, contrary to T-FP, NFSP does not exploit the threshold structures expressed in Theorem 4.1 and as a result is more complex. HSVI is a state-of-the-art dynamic programming algorithm for one-sided POSGs.

Discussion of the evaluation results. Fig. 5 shows the learning curves of the strategies obtained during the T-FP self-play process. The red curve represents the results from the simulation system and the blue curves show the results from the emulation system. The purple and orange curves give the performance of the Snort IPS baseline and the baseline strategy that mandates a stop action whenever an IPS alert occurs, respectively. The dashed black curve gives the performance of the baseline strategy that assumes knowledge of the exact intrusion time.

The results in Fig. 5 lead us to the following conclusions. First, the fact that all learning curves seem to converge suggests to us that the learned strategies have converged as well. Second, we observe that the exploitability of the learned strategies converges to small values (left plot of Fig. 5). This indicates that the learned strategies approximate a Nash equilibrium both in the simulation system and in the emulation system. Third, we see from the middle plot in Fig. 5 that both baseline strategies show decreasing performance as the attacker updates its strategy. In contrast, the learned defender strategy improves its performance over time. This shows the benefit of using a game-theoretic approach, whereby the defender's strategy is optimized against a dynamic attacker.

Fig. 6 allows a comparison between T-FP and the two baseline algorithms (NFSP and HSVI) as they execute in

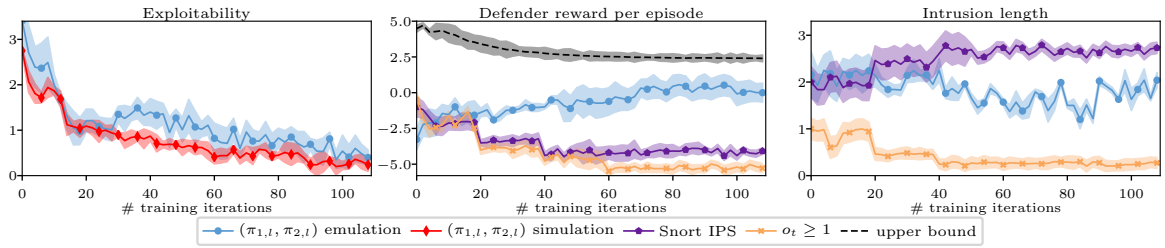

Figure 5. Learning curves from the self-play process with T-FP; the red curve shows simulation results and the blue curves show emulation results; the purple, orange, and black curves relate to baseline strategies; the figures show different performance metrics: exploitability, episodic reward, and the length of intrusion; the curves indicate the mean and the 95% confidence interval over four training runs with different random seeds.

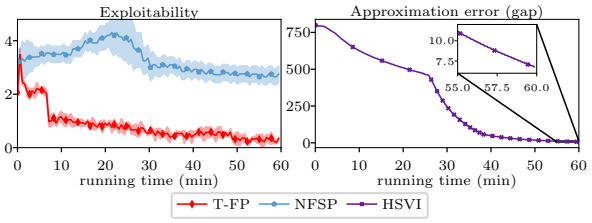

Figure 6. Comparison between T-FP and two baseline algorithms: NFSP and HSVI; all curves show simulation results; the red curve relates to T-FP; the blue curve to NFSP; the purple curve to HSVI; the left plot shows the exploitability metric and the right plot shows the HSVI approximation error; the curves depicting T-FP and NFSP show the mean and the 95% confidence interval over four training runs with different random seeds.

the simulation system. Since T-FP and NFSP both implement fictitious self-play, they allow for a direct comparison. We observe that T-FP converges much faster to a Nash equilibrium than NFSP. We expect the fast convergence of T-FP due to its design to exploit structural properties of the stopping game.

The right plot of Fig. 6 shows that HSVI reaches an HSVI approximation error < 5 within an hour. We expected slower convergence due to findings in (Horák, 2019). A direct comparison between T-FP and HSVI is not possible due to their different nature.

Additional results can be found in (Hammar & Stadler, 2022c).

7. Related Work

Related works on finding security strategies through reinforcement learning include the simulation-based studies in (Ridley, 2018; Blum, 2021; Hammar & Stadler, 2020; Tran et al., 2021; Schwartz et al., 2020; Dhir et al., 2021), the emulation-based studies in (Akbari et al., 2020; Liu et al., 2018; Aydeger et al., 2021; Hammar & Stadler, 2022b), and the papers (Standen et al.; Molina-Markham et al., 2021; Hammar & Stadler, 2022d; Li et al., 2021), which describe ongoing efforts in building emulation platforms for reinforcement learning research in the cyber domain.

Game-theoretic formulations based on optimal stopping theory can be found in prior research on Dynkin games (Dynkin, 1969) and a similar game model to ours is FlipIt (van Dijk et al., 2013).

A review of the related work is available in the extended arXiv version of this paper (Hammar & Stadler, 2022c).

8. Conclusion and Future Work

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

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

We plan to extend this work by combining our model for when to take actions with a model for action selection.

References

Akbari, I., Tahoun, E., Salahuddin, M. A., Limam, N., and Boutaba, R. Atmos: Autonomous threat mitigation in sdn using reinforcement learning. In *NOMS IEEE/IFIP Network Operations and Management Symposium*, pp. 1–9, 2020. doi: 10.1109/NOMS47738.2020.9110426.

Aydeger, A., Manshaei, M. H., Rahman, M. A., and Akkaya, K. Strategic defense against stealthy link flooding attacks: A signaling game approach. *IEEE Transactions on Network Science and Engineering*, 8(1):751–764, 2021.

Bellman, R. A markovian decision process. *Journal of Mathematics and Mechanics*, 6(5):679–684, 1957.

Blum, W. Gamifying machine learning for stronger security and ai models, 2021.

Brown, G. W. Iterative solution of games by fictitious play, 1951. Activity analysis of production and allocation.

Dhir, N. et al. Prospective artificial intelligence approaches for active cyber defence. 2021.

Dynkin, E. A game-theoretic version of an optimal stopping problem. *Dokl. Akad. Nauk SSSR*, 385:16–19, 1969.

Fuchsberger, A. Intrusion detection systems and intrusion prevention systems. *Inf. Secur. Tech. Rep.*, 10(3), 2005.

Hammar, K. and Stadler, R. Finding effective security strategies through reinforcement learning and Self-Play. In *International Conference on Network and Service Management (CNSM 2020)*, Izmir, Turkey, 2020.

Hammar, K. and Stadler, R. Learning intrusion prevention policies through optimal stopping. In *International Conference on Network and Service Management (CNSM 2021)*, Izmir, Turkey, 2021.

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

Hammar, K. and Stadler, R. Intrusion prevention through optimal stopping. *IEEE Transactions on Network and Service Management*, pp. 1–1, 2022b. doi: 10.1109/TNSM.2022.3176781.

Hammar, K. and Stadler, R. Learning security strategies through game play and optimal stopping, 2022c. URL https://arxiv.org/abs/2205.14694.

Hammar, K. and Stadler, R. A system for interactive examination of learned security policies. In *NOMS IEEE/IFIP Network Operations and Management Symposium*, 2022d.

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

Horák, K. *Scalable Algorithms for Solving Stochastic Games with Limited Partial Observability*. PhD thesis, 2019.

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

Li, L., Fayad, R., and Taylor, A. Cygil: A cyber gym for training autonomous agents over emulated network systems. *CoRR*, abs/2109.03331, 2021.

Liu, Y. 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*, pp. 1–6, 2018.

Molina-Markham, A., Miniter, C., Powell, B., and Ridley, A. Network environment design for autonomous cyberdefense. 2021.

Nash, J. F. Non-cooperative games. *Annals of Mathematics*, 54:286–295, 1951.

Puterman, M. L. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley and Sons, Inc., USA, 1st edition, 1994. ISBN 0471619779.

Ridley, A. Machine learning for autonomous cyber defense, 2018. The Next Wave, Vol 22, No.1 2018.

Schwartz, J., Kurniawati, H., and El-Mahassni, E. Pomdp + information-decay: Incorporating defender's behaviour in autonomous penetration testing. *Proceedings of the International Conference on Automated Planning and Scheduling*, 30:235–243, Jun. 2020.

Standen, M., Lucas, M., Bowman, D., Richer, T. J., Kim, J., and Marriott, D. Cyborg: A gym for the development of autonomous cyber agents.

Tran, K., Akella, A., Standen, M., Kim, J., Bowman, D., Richer, T., and Lin, C.-T. Deep hierarchical reinforcement agents for automated penetration testing, 2021.

van Dijk, M., Juels, A., Oprea, A., and Rivest, R. L. Flipit: The game of "stealthy takeover". *Journal of Cryptology*, (4), Oct 2013. ISSN 1432-1378.

von Neumann, J. Zur Theorie der Gesellschaftsspiele. (German) [On the theory of games of strategy]. 100:295–320, 1928. ISSN 0025-5831 (print), 1432-1807 (electronic).

Wald, A. *Sequential Analysis*. Wiley and Sons, 1947.