Scalable Learning of Intrusion Responses through Recursive Decomposition

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
June 18, 2023

***Abstract*—We study automated intrusion response and formulate the interaction between an attacker and a defender on an IT infrastructure as a partially observed stochastic game where attack and defense strategies evolve through reinforcement learning and self-play. Direct application of reinforcement learning to any non-trivial instantiation of this game is impractical due to the exponential growth of the computational complexity with the size of the infrastructure. We propose a decompositional approach to deal with this challenge and prove that under assumptions generally met in practice, the game decomposes recursively into subgames that can be solved in parallel. We further show that the optimal defender strategies exhibit threshold structures. To solve the decomposed game we develop Decompositional Fictitious Self-Play (DFSP), an efficient self-play algorithm that learns Nash equilibria through stochastic approximation. We evaluate the learned strategies in a virtual infrastructure in which we run real intrusions and response actions. The results show that the learned strategies are near-optimal and that DFSP outperforms a state-of-the-art algorithm.**

***Index Terms*—Cybersecurity, network security, intrusion response, decomposition, reinforcement learning, game theory, Markov decision process, optimal control, digital twin, MDP.**

I. Introduction

An organization's security strategy has traditionally been defined, implemented, and updated by domain experts. This approach can provide basic security for an organization's communication and computing infrastructure. As update cycles become shorter and attacks increase in sophistication, meeting the security requirements becomes increasingly difficult. To address this challenge, significant efforts to automate the process of obtaining security strategies have started [2]–[28].

A promising direction of recent research is to automatically find security strategies through reinforcement learning methods, whereby the problem of finding effective strategies is modeled as a Markov decision problem and strategies are learned in simulation (see survey [29]). While encouraging results have been obtained following this approach (see e.g. [16] and [21]), key challenges remain. 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 with static attackers and it is not clear how they generalize to real systems with dynamic attackers.

In our previous work, we have developed a game-theoretic framework that addresses the above limitations and allows to automatically learn near-optimal defender strategies against dynamic attackers (see Fig. 1) [16]–[18]. This framework centers around a *digital twin* of the targeted infrastructure, which we use to run attack scenarios and defender responses. Such runs produce system measurements and logs, from which we estimate infrastructure statistics. We then use the estimated statistics to instantiate simulations of the infrastructure's dynamics and learn defender strategies through reinforcement learning, whose performances we assess in the digital twin. This closed-loop process executes iteratively and provides continuously evolving and improving defender strategies for the target infrastructure.

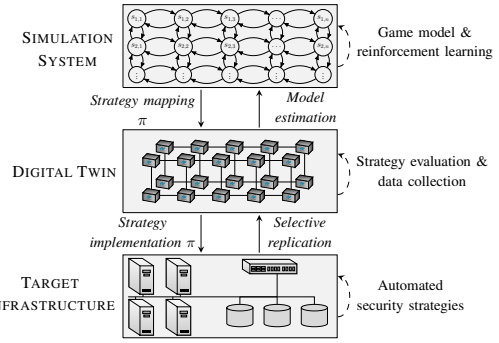


Fig. 1: Our game-theoretic framework for finding and evaluating intrusion response strategies [16]–[18].

In this paper, we apply our framework to an *intrusion response* use case that involves the IT infrastructure of an organization (see Fig. 2). We formulate this use case as a partially observed stochastic game between two players – the operator of the infrastructure, which we call the defender; and an attacker, which seeks to intrude on the infrastructure. A key issue in the design of this game is the level of abstraction at which the infrastructure is modeled. The more detailed we construct the model the closer it can capture reality. This however comes at the expense of high computational complexity. To manage this complexity, we recursively decompose the game into simpler subgames, which allows detailed modeling of the infrastructure while keeping the computational complexity low.

The decomposition involves three steps. First, we exploit that many IT infrastructures can be partitioned into workflows

that are isolated from each other. We can therefore decompose the game into *independent subgames* (one per workflow) that can be solved in parallel. Second, we show that workflows tend to have topological structures that allow to decompose them into component subgames with *optimal substructure* [30, Ch. 15]. This means that a solution of the original game can be obtained from solutions of the subgames. Third, we prove that the problems of selecting *which* response action to apply to each component and selecting *when* to apply it can be separated, which allows efficient computation and theoretical insight through *optimal stopping* theory [31].

Based on the above properties, we develop an efficient reinforcement learning algorithm, called Decompositional Fictitious Self-Play (DFSP), which allows scalable approximation of near-optimal defender strategies for IT infrastructures.

In summary, we make the following contributions.

(i) We formulate the intrusion response problem as a partially observed stochastic game and prove that under assumptions generally met in practice, the game decomposes into subgames that can be solved in parallel.
(ii) We design DFSP, an efficient reinforcement learning algorithm for approximating Nash equilibria of the decomposed game.
(iii) We evaluate the learned response strategies against real network intrusions in a virtual infrastructure.

II. Related Work

It is widely appreciated that networked systems typically found in engineering, physics, and biology frequently exhibit modular structure in their connection topology, and that this structure can be exploited in the design of control algorithms [32], [33]. System decomposition for automatic control was first suggested by Šiljak in 1978 [34] and decompositional approaches such as divide and conquer, layering, and hierarchy-structures are also well-established in the design of large-scale systems, a notable example being the Internet [35]. Similar decompositions are also frequently used to deal with problems in robotics and multi-agent systems, where a famous example is the subsumption architecture [36]. Within the fields of decision- and game-theory, decomposition is studied under the umbrellas of factored decision processes [37]–[40] and factored games [41], [42].

Decomposition as a means to automate intrusion responses is first studied in [41], [43]–[45]. The work in [41] formulates the interaction between a defender and an attacker on a cyber-physical infrastructure as a factored Markov game and develops a decomposition based on linear programming. Following a similar approach, the work in [44] studies a Markov game formulation and shows that a multi-stage game can be decomposed into a sequence of one-stage games. In another line of work, [43] models intrusion response as a minimax control problem and develops a heuristic decomposition based on clustering and influence graphs. This approach resembles the work in [45], which studies a factored decision process and proposes a hierarchical decomposition.

All of the above references find that decomposition is key to obtain effective strategies for large-scale infrastructures.

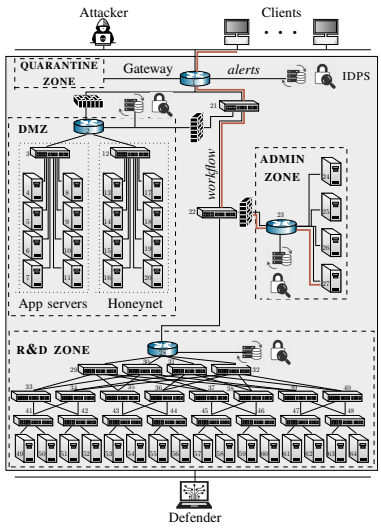


Fig. 2: The target IT infrastructure and the actors involved in the intrusion response use case.

Despite these encouraging findings, none of the referenced works have demonstrated their solution outside of simulation. Furthermore, most of the existing works assume full observability and do not consider reinforcement learning methods.

For a comprehensive review of previous work that study automated intrusion response (including works that do not use decomposition), see [17, §VII].

III. 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 (see Fig. 2). The infrastructure is segmented into *zones* with servers that run network services. Services are realized by *workflows* that are accessed by clients through a gateway, which also is open to the attacker.

The attacker's goal is to intrude on the infrastructure, compromise servers, and disrupt workflows. It can take three types of actions to achieve this goal: (*i*) reconnaissance; (*ii*) brute-force attacks; and (*iii*) exploits (see Fig. 3).

The defender continuously monitors the infrastructure through accessing and analyzing intrusion detection alerts and other statistics. It can take four types of defensive actions to respond to possible intrusions: (*i*) migrate components between zones; (*ii*) redirect or block network flows; (*iii*) shut down components; and (*iv*) revoke access to components (see Fig. 4). When deciding on defensive actions, the defender balances two objectives: a) maintain workflows to clients; and b) respond to possible intrusions while minimizing costs.

IV. Formalizing the Intrusion Response Problem

We formalize the above use case as an optimization problem where the goal is to select an optimal sequence of defender actions in response to a sequence of attacker actions. We assume a dynamic attacker, which leads to a game model of

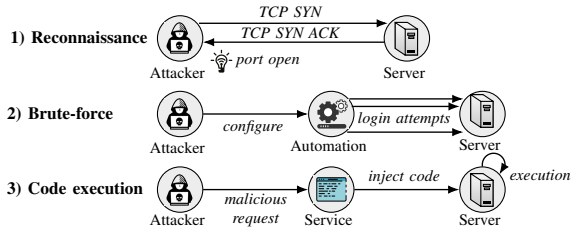


Fig. 3: Attacker actions: (*i*) reconnaissance actions; (*ii*) brute-force attacks; and (*iii*) code execution attacks.

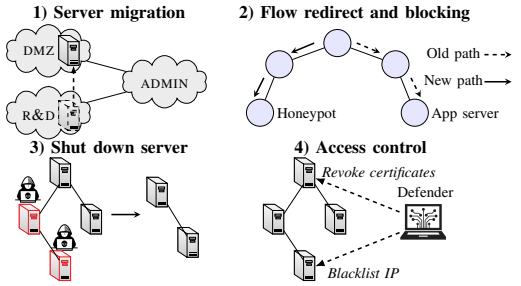


Fig. 4: Defender actions: (*i*) migrate a server between two zones; (*ii*) redirect or block traffic flows to a server; (*iii*) shut down a server; and (*iv*) revoke access to a server.

the intrusion response problem. The game is played on the IT infrastructure, which we model as a discrete-time dynamical system whose evolution depends on the actions by the attacker and the defender. Both actors have partial observability of the system state and their observations depend on traffic generated by clients requesting service, which we assume can be described by a stationary process.

Notations. Boldface lower case letters (e.g. $\mathbf{x}$) denote row vectors and upper case calligraphic letters (e.g. $\mathcal{V}$) represent sets. The set of probability distributions over $\mathcal{V}$ is denoted with $\Delta(\mathcal{V})$. A random variable is denoted with upper case (e.g. X) and a random vector is denoted with boldface (e.g. $\mathbf{X} = (X_1, \ldots, X_n)$). $\mathbb{P}$ is the probability measure and the expectation of f with respect to X is denoted with $\mathbb{E}_X[f]$. When f includes many variables that depend on π we simplify the notation to $\mathbb{E}_\pi[f]$. We use $x \sim f$ to denote that x is sampled from f and write $\mathbb{P}[x|z, y]$ instead of $\mathbb{P}[X = x|Z = z, Y = y]$ when X, Z, Y are clear from the context.

A. Modeling the Infrastructure and Services

Following the description in §III, we consider an IT infrastructure with application servers connected by a communication network that is segmented into zones (see Fig. 2). Overlaid on this physical infrastructure is a virtual infrastructure that includes *nodes*, which collectively offer services to clients.

A service is modeled as a *workflow*, which comprises a set of interdependent nodes. A dependency between two nodes reflects information exchange through service invocations. We assume that each node belongs to exactly one workflow. As an example of a virtual infrastructure, we can think of a microservice architecture where a workflow is defined as a chain of microservices (see Fig. 5).

Infrastructure. We model the virtual infrastructure as a finite directed graph $\mathcal{G} \triangleq \langle \{\text{gw}\} \cup \mathcal{V}, \mathcal{E} \rangle$. The graph has a tree structure and is rooted at the gateway gw. Each node $i \in \mathcal{V}$ has three state variables: $v_{i,t}^{(\text{Z})}$ is a realization of the random variable $V_{i,t}^{(\text{Z})}$ and represents the zone in which the node resides; $v_{i,t}^{(\text{R})}$ is a realization of $V_{i,t}^{(\text{R})}$ and represents the reconnaissance state; and $v_{i,t}^{(\text{I})}$ is a realization of $V_{i,t}^{(\text{I})}$, which represents the intrusion state. $V_{i,t}^{(\text{R})}$ and $V_{i,t}^{(\text{I})}$ are binary random variables where $V_{i,t}^{(\text{R})} = 1$ and $V_{i,t}^{(\text{I})} = 1$ mean that i have been *discovered* and *compromised* by the attacker, respectively. We call a node *active* if it is functional as part of a workflow. Due to defender actions a node $i \in \mathcal{V}$ may become inactive (e.g. it may be shut down), which we model by changing $v_{i,t}^{(\text{Z})}$.

Workflows. We model a workflow $\mathbf{w} \in \mathcal{W}$ as a subtree of the infrastructure graph. Workflows do not overlap except for the gateway which belongs to all workflows.

B. Modeling Actors

The intrusion response use case involves three types of actors: clients, an attacker, and a defender (see Fig. 2).

Clients. Clients consume services of the infrastructure by accessing workflows. We model client behavior through stationary stochastic processes, which affect the observations available to the attacker and the defender.

Attacker. At each time t, the attacker takes an action $\mathbf{a}_t^{(\text{A})}$, which is defined as the composition of the local actions on all nodes $\mathbf{a}_t^{(\text{A})} \triangleq (\mathbf{a}_{1,t}^{(\text{A})}, \ldots, \mathbf{a}_{|\mathcal{V}|,t}^{(\text{A})}) \in \mathcal{A}_\text{A}$, where $\mathcal{A}_\text{A}$ is finite. A local action can be a null action (denoted with $\perp$) or an offensive action (see examples in Fig. 3). An offensive action on a node i may change the reconnaissance state $v_{i,t}^{(\text{R})}$ or the intrusion state $v_{i,t}^{(\text{I})}$. A node i can only be compromised if it is discovered, i.e. if $v_{i,t}^{(\text{R})} = 1$. We denote this constraint with $\mathbf{a}_t^{(\text{A})} \in \mathcal{A}_\text{A}(\mathbf{s}_t)$.

The attacker state $\mathbf{S}_t^{(\text{A})} \triangleq \big(V_{i,t}^{(\text{I})}, V_{i,t}^{(\text{R})}\big)_{i \in \mathcal{V}}$ evolves as

$$\mathbf{s}_{t+1}^{(\text{A})} \sim f_\text{A}\big(\cdot \mid \mathbf{S}_t^{(\text{A})}, \mathbf{A}_t^{(\text{A})}, \mathbf{A}_t^{(\text{D})}\big) \tag{1}$$

where $\mathbf{A}_t^{(\text{A})}$ and $\mathbf{A}_t^{(\text{D})}$ are random vectors (with realizations $\mathbf{a}_t^{(\text{A})}$ and $\mathbf{a}_t^{(\text{D})}$) representing the actions at time t of the attacker and the defender.

Defender. At each time t, the defender takes an action $\mathbf{a}_t^{(\text{D})}$, which is defined as the composition of the local actions on all nodes $\mathbf{a}_t^{(\text{D})} \triangleq (\mathbf{a}_{1,t}^{(\text{D})}, \ldots, \mathbf{a}_{|\mathcal{V}|,t}^{(\text{D})}) \in \mathcal{A}_\text{D}$, where $\mathcal{A}_\text{D}$ is finite. A local action can be a defensive action or the null action $\perp$ (see examples in Fig. 4). Each defensive action $\mathbf{a}_i^{(\text{D})} \neq \perp$ sets $\mathbf{s}_i^{(\text{A})}$ to $(0, 0)$ and may change $v_i^{(\text{Z})}$. The defender state $\mathbf{s}_t^{(\text{D})} \triangleq \big(v_{i,t}^{(\text{Z})}\big)_{i \in \mathcal{V}}$ evolves according to

$$\mathbf{s}_{t+1}^{(\text{D})} \sim f_\text{D}\big(\cdot \mid \mathbf{S}_t^{(\text{D})}, \mathbf{A}_t^{(\text{D})}\big) \tag{2}$$

C. Observability and Strategies

At each time t, the defender and the attacker both observe $\mathbf{o}_t \triangleq \big(\mathbf{o}_{1,t}, \ldots, \mathbf{o}_{|\mathcal{V}|,t}\big) \in \mathcal{O}$, where $\mathcal{O}$ is finite. (In our use case $\mathbf{o}_t$ relates to the number of IDPS alerts per node.) $\mathbf{o}_t$

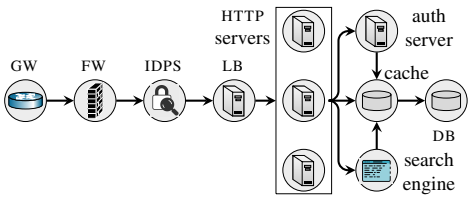


Fig. 5: Dependency graph of a workflow consisting of a chain of virtual network functions and microservices; FW, LB, and IDPS are acronyms for firewall, load balancer, and intrusion detection and prevention system, respectively.

is drawn from the random vector $\mathbf{O}_t \triangleq (\mathbf{O}_{1,t}, \ldots, \mathbf{O}_{|\mathcal{V}|,t})$ whose marginal distributions $Z_{\mathbf{O}_1}, \ldots, Z_{\mathbf{O}_{|\mathcal{V}|}}$ are conditionally independent given $\mathbf{S}^{(\mathrm{D})}_{i,t+1}$ and $\mathbf{S}^{(\mathrm{A})}_{i,t+1}$. As a consequence, the joint conditional distribution Z is given by

$$Z\big(\mathbf{O}_{t+1} = \mathbf{o} \mid \mathbf{S}_{t+1}\big) = \prod_{i=1}^{|\mathcal{V}|} Z_{\mathbf{O}_i}\big(\mathbf{O}_{i,t+1} = \mathbf{o}_i \mid \mathbf{S}_{i,t+1}\big) \quad (3)$$

where $\mathbf{o} \in \mathcal{O}$ and $\mathbf{S}_{i,t+1} \triangleq (\mathbf{S}^{(\mathrm{D})}_{i,t+1}, \mathbf{S}^{(\mathrm{A})}_{i,t+1})$.

The sequence of observations and states at times $1, \ldots, t$ forms the histories $\mathbf{h}^{(\mathrm{D})}_t \in \mathcal{H}_\mathrm{D}$ and $\mathbf{h}^{(\mathrm{A})}_t \in \mathcal{H}_\mathrm{A}$. These histories are realizations of the random vectors $\mathbf{H}^{(\mathrm{D})}_t \triangleq (\mathbf{S}^{(\mathrm{D})}_1, \mathbf{A}^{(\mathrm{D})}_1, \mathbf{O}_1, \ldots, \mathbf{A}^{(\mathrm{D})}_{t-1}, \mathbf{S}^{(\mathrm{D})}_t, \mathbf{O}_t)$ and $\mathbf{H}^{(\mathrm{A})}_t \triangleq (\mathbf{S}^{(\mathrm{A})}_1, \mathbf{A}^{(\mathrm{A})}_1, \mathbf{O}_1, \ldots, \mathbf{A}^{(\mathrm{A})}_{t-1}, \mathbf{S}^{(\mathrm{A})}_t, \mathbf{O}_t)$. Based on their respective histories, the defender and the attacker select actions, which define the defender strategy $\pi_\mathrm{D} \in \Pi_\mathrm{D} : \mathcal{H}_\mathrm{D} \to \Delta(\mathcal{A}_\mathrm{D})$ and the attacker strategy $\pi_\mathrm{A} \in \Pi_\mathrm{A} : \mathcal{H}_\mathrm{A} \to \Delta(\mathcal{A}_\mathrm{A})$.

D. The Intrusion Response Problem

When selecting the strategy π_D the defender must balance two conflicting objectives: maximize the workflow utility towards its clients and minimize the cost of intrusion. The weight $\eta \in \mathbb{R}$ captures the trade-off between these two objectives, which results in the bi-objective function

$$J \triangleq \sum_{t=1}^{\infty} \gamma^{t-1} \Bigg(\sum_{\mathbf{w} \in \mathcal{W}} \sum_{i \in \mathcal{V}_\mathbf{w}} \underbrace{\eta u^{(\mathrm{w})}_{i,t}}_{\text{workflows utility}} - \underbrace{v^{(\mathrm{I})}_{i,t}}_{\text{intrusion cost}} \Bigg) \quad (4)$$

where $\gamma \in [0, 1)$ is a discount factor and $u^{(\mathrm{w})}_{i,t}$ expresses the workflow utility associated with node i at time t. We assume that $u^{(\mathrm{w})}_{i,t}$ is a bounded function, dependent on the action $\mathbf{a}^{(\mathrm{D})}_{i,t}$ and on the number of active nodes in the subtree rooted at i (denoted with $l_{i,t}$).

Given (4) and an attacker strategy π_A, the intrusion response problem can be stated as

$$\begin{aligned}
\underset{\pi_\mathrm{D} \in \Pi_\mathrm{D}}{\text{maximize}} \quad & \mathbb{E}_{(\pi_\mathrm{D}, \pi_\mathrm{A})}[J] && && (5a)\\
\text{subject to} \quad & \mathbf{s}^{(\mathrm{D})}_{t+1} \sim f_\mathrm{D}\big(\cdot \mid \mathbf{S}^{(\mathrm{D})}_t, \mathbf{A}^{(\mathrm{D})}_t\big) && \forall t \quad (5b)\\
& \mathbf{s}^{(\mathrm{A})}_{t+1} \sim f_\mathrm{A}\big(\cdot \mid \mathbf{S}^{(\mathrm{A})}_t, \mathbf{A}^{(\mathrm{A})}_t, \mathbf{A}^{(\mathrm{D})}_t\big) && \forall t \quad (5c)\\
& \mathbf{o}_{t+1} \sim Z\big(\cdot \mid \mathbf{S}^{(\mathrm{D})}_{t+1}, \mathbf{S}^{(\mathrm{A})}_{t+1}\big) && \forall t \quad (5d)\\
& \mathbf{a}^{(\mathrm{A})}_t \sim \pi_\mathrm{A}\big(\cdot \mid \mathbf{H}^{(\mathrm{A})}_t\big) && \forall t \quad (5e)\\
& \mathbf{a}^{(\mathrm{D})}_t \sim \pi_\mathrm{D}\big(\cdot \mid \mathbf{H}^{(\mathrm{D})}_t\big) && \forall t \quad (5f)
\end{aligned}$$

where $\mathbb{E}_{(\pi_\mathrm{D}, \pi_\mathrm{A})}$ denotes the expectation of the random vectors $(\mathbf{H}^{(\mathrm{D})}_t, \mathbf{H}^{(\mathrm{A})}_t)_{t \in \{1,2,\ldots\}}$ under the strategy profile $(\pi_\mathrm{D}, \pi_\mathrm{A})$; (5b)–(5c) are the dynamics constraints; (5d) describes the observations; and (5e)–(5f) capture the actions.

Solving (5) yields an optimal defender strategy against a *static* attacker with a fixed strategy. Note that this defender strategy is generally not optimal against a different attacker strategy. For this reason, we aim to find a defender strategy that maximizes the minimum value of J (4) across all possible attacker strategies. This objective can be formally expressed as a maxmin problem [46]:

$$\underset{\pi_\mathrm{D} \in \Pi_\mathrm{D}}{\text{maximize}} \; \underset{\pi_\mathrm{A} \in \Pi_\mathrm{A}}{\text{minimize}} \quad \mathbb{E}_{(\pi_\mathrm{D}, \pi_\mathrm{A})}[J] \quad \text{subject to (5b)–(5f)} \quad (6)$$

Solving (6) corresponds to finding a Nash equilibrium [47, Eq. 1] and can be analyzed through game theory.

V. The Intrusion Response Game

The maxmin problem in (6) defines a stationary, finite, and zero-sum Partially Observed Stochastic Game with Public Observations (a PO-POSG) [48, Def. 1]:

$$\Gamma = \langle \mathcal{N}, (\mathcal{S}_\mathrm{k}, \mathcal{A}_\mathrm{k}, f_\mathrm{k}, \mathbf{b}^{(\mathrm{k})}_1)_{\mathrm{k} \in \mathcal{N}}, u, \gamma, \mathcal{O}, Z \rangle \quad (7)$$

The game Γ has two players $\mathcal{N} = \{\mathrm{D}, \mathrm{A}\}$ with D being the defender and A being the attacker. $(\mathcal{S}_\mathrm{k})_{\mathrm{k} \in \mathcal{N}}$ defines the states, $(\mathcal{A}_\mathrm{k})_{\mathrm{k} \in \mathcal{N}}$ defines the actions, and $\mathcal{O}$ defines the observations. The transition functions $(f_\mathrm{k})_{\mathrm{k} \in \mathcal{N}}$ are defined by (5b)–(5c), the observation function Z is defined in (3), and the utility function u is the expression within brackets in (4). $(\mathbf{b}^{(\mathrm{k})}_1)_{\mathrm{k} \in \mathcal{N}}$ are the state distributions at $t = 1$ and γ is the discount factor in (4).

Game play. When the game starts at $t = 1$, $\mathbf{s}^{(\mathrm{D})}_1$ and $\mathbf{s}^{(\mathrm{A})}_1$ are sampled from $\mathbf{b}^{(\mathrm{D})}_1$ and $\mathbf{b}^{(\mathrm{A})}_1$. A play of the game proceeds in time-steps $t = 1, 2, \ldots$. At each time t, the defender observes $\mathbf{h}^{(\mathrm{D})}_t$ and the attacker observes $\mathbf{h}^{(\mathrm{A})}_t$. Based on these observations, both players select actions according to their respective strategies, i.e. $\mathbf{a}^{(\mathrm{D})}_t \sim \pi_\mathrm{D}(\cdot \mid \mathbf{h}^{(\mathrm{D})}_t)$ and $\mathbf{a}^{(\mathrm{A})}_t \sim \pi_\mathrm{A}(\cdot \mid \mathbf{h}^{(\mathrm{A})}_t)$. As a result of these actions, five events occur at time $t+1$: (*i*) $\mathbf{o}_{t+1}$ is sampled from Z; (*ii*) $\mathbf{s}^{(\mathrm{D})}_{t+1}$ is computed using f_D; (*iii*) $\mathbf{s}^{(\mathrm{A})}_{t+1}$ is computed using f_A; (*iv*) the defender receives the utility u_t; and (*v*) the attacker receives the utility $-u_t$.

Belief states. Based on their histories $\mathbf{h}^{(\mathrm{D})}_t$ and $\mathbf{h}^{(\mathrm{A})}_t$, both players form beliefs about the unobservable components of the state $\mathbf{s}_t$, which are expressed through the belief states $\mathbf{b}^{(\mathrm{D})}_t(\mathbf{s}^{(\mathrm{A})}_t) \triangleq \mathbb{P}[\mathbf{s}^{(\mathrm{A})}_t | \mathbf{H}^{(\mathrm{D})}_t]$ and $\mathbf{b}^{(\mathrm{A})}_t(\mathbf{s}^{(\mathrm{D})}_t) \triangleq \mathbb{P}[\mathbf{s}^{(\mathrm{D})}_t | \mathbf{H}^{(\mathrm{A})}_t]$, which are realizations of $\mathbf{B}^{(\mathrm{D})}_t$ and $\mathbf{B}^{(\mathrm{A})}_t$. At each time $t > 1$ the beliefs are updated via [48, Eq. 1]. The initial beliefs at $t = 1$ are the degenerate distributions $\mathbf{b}^{(\mathrm{D})}_1(\mathbf{0}_{2|\mathcal{V}|}) = 1$ and $\mathbf{b}^{(\mathrm{A})}_1(\mathbf{s}^{(\mathrm{D})}_1) = 1$.

Best response strategies. A defender strategy $\tilde{\pi}_\mathrm{D} \in \Pi_\mathrm{D}$ is called a *best response* against $\pi_\mathrm{A} \in \Pi_\mathrm{A}$ if it *maximizes* J (4). Similarly, an attacker strategy $\tilde{\pi}_\mathrm{A}$ is called a best response against π_D if it *minimizes* J (4). Hence, the best response correspondences are

$$B_\mathrm{D}(\pi_\mathrm{A}) \triangleq \arg\max_{\pi_\mathrm{D} \in \Pi_\mathrm{D}} \mathbb{E}_{(\pi_\mathrm{D}, \pi_\mathrm{A})}[J] \quad (8)$$

$$B_{\mathrm{A}}(\pi_{\mathrm{D}}) \triangleq \arg\min_{\pi_{\mathrm{A}} \in \Pi_{\mathrm{A}}} \mathbb{E}_{(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})}[J] \tag{9}$$

Optimal strategies. An optimal defender strategy π_{D}^* is a best response strategy against any attacker strategy that *minimizes* J. Similarly, an optimal attacker strategy π_{A}^* is a best response against any defender strategy that *maximizes* J. Hence, when both players follow optimal strategies, they play best response strategies against each other:

$$(\pi_{\mathrm{D}}^*, \pi_{\mathrm{A}}^*) \in B_{\mathrm{D}}(\pi_{\mathrm{A}}^*) \times B_{\mathrm{A}}(\pi_{\mathrm{D}}^*) \tag{10}$$

Since no player has an incentive to change its strategy, $(\pi_{\mathrm{D}}^*, \pi_{\mathrm{A}}^*)$ is a Nash equilibrium [47, Eq. 1].

We know from game theory that Γ has a mixed Nash equilibrium [47]–[50] and we know from Markov decision theory that $B_{\mathrm{D}}(\pi_{\mathrm{A}})$ and $B_{\mathrm{A}}(\pi_{\mathrm{D}})$ are non-empty [31], [51]. Based on these standard results, we state the following theorem.

Theorem 1.

(A) The game Γ (7) with instantiation described in §IV has a mixed Nash equilibrium.

(B) The best response correspondences (8)–(9) satisfy $|B_{\mathrm{D}}(\pi_{\mathrm{A}})| > 0$ and $|B_{\mathrm{A}}(\pi_{\mathrm{D}})| > 0 \; \forall (\pi_{\mathrm{A}}, \pi_{\mathrm{D}}) \in \Pi_{\mathrm{A}} \times \Pi_{\mathrm{D}}$.

Proof. The statement in (A) follows from the following sufficient conditions: (*i*) Γ is stationary, finite, and zero-sum; (*ii*) Γ has public observations; and (*iii*) $\gamma \in [0, 1)$. Due to these conditions, the existence proofs in [49, §3], [50, Thm. 2.3], and [48, Thm. 1] apply, which show that Γ can be modeled as a finite strategic game, for which Nash's theorem applies [47, Thm. 1]. In the interest of space we do not restate the proof.

To prove (B), we note that obtaining a pair of best response strategies $(\tilde{\pi}_{\mathrm{D}}, \tilde{\pi}_{\mathrm{A}}) \in B_{\mathrm{D}}(\pi_{\mathrm{A}}) \times B_{\mathrm{A}}(\pi_{\mathrm{D}})$ for a given strategy pair $(\pi_{\mathrm{A}}, \pi_{\mathrm{D}}) \in \Pi_{\mathrm{A}} \times \Pi_{\mathrm{D}}$ amounts to solving two finite and stationary POMDPs (Partially Observed Markov Decision Processes) with discounted utilities [50]. It then follows from Markov decision theory that a pair of pure best response strategies $(\tilde{\pi}_{\mathrm{D}}, \tilde{\pi}_{\mathrm{A}})$ exists [51, Thm. 6.2.7] [31, Thm. 7.6.1-7.6.2]. For the sake of brevity we do not restate the proof, which is based on Banach's fixed-point theorem [52, Thm. 6, p. 160]. □

VI. Decomposing the Intrusion Response Game

In this section we present the main contribution of the paper. We show how the game Γ (7) with the instantiation described in §IV can be recursively decomposed into subgames with optimal substructure [30, Ch. 15], which means that a solution of the original game can be obtained from solutions of the subgames. We further show that the subgames can be solved in parallel and that their space complexities are independent of the number of nodes $|\mathcal{V}|$. Note that the complexity of the original game increases exponentially with $|\mathcal{V}|$ (see Fig. 6).

Theorem 2 (Decomposition theorem)**.**

(A) The game Γ (7) with the instantiation described in §IV can be decomposed into independent workflow subgames $\Gamma^{(\mathbf{w}_1)}, \ldots, \Gamma^{(\mathbf{w}_{|\mathcal{W}|})}$. Due to their independence, the subgames have optimal substructure.

(B) Each subgame $\Gamma^{(\mathbf{w})}$ can be further decomposed into node subgames $(\Gamma^{(i,l_i)})_{i \in \mathcal{V}_{\mathbf{w}}, l_i \in \{1,\ldots,|\mathcal{K}_i|\}}$, where $\mathcal{K}_i$ is the set of nodes in the subtree of the infrastructure graph $\mathcal{G}$ rooted i. The subgames have optimal substructure and space complexities independent of $|\mathcal{V}|$.

(C) For each subgame $\Gamma^{(i,l_i)}$, a best response strategy for the defender can be characterized by a switching curve, under the assumption that the distributions $Z_{\mathbf{O}_1|\mathbf{s}^{(\mathrm{A})}}, \ldots, Z_{\mathbf{O}_{|\mathcal{V}|}|\mathbf{s}^{(\mathrm{A})}}$ (3) are totally positive of order 2 (i.e. TP-2 [31, Def. 10.2.1]).

Notation(s)	*Description*
$\mathcal{G}, \mathcal{G}_{\mathbf{w}}$	Infrastructure graph and subtree for workflow $\mathbf{w}$
$\mathcal{V}, \mathcal{E}$	Sets of nodes and edges in $\mathcal{G}$
$\mathcal{V}_{\mathbf{w}}, \mathcal{E}_{\mathbf{w}}$	Sets of nodes and edges in $\mathcal{G}_{\mathbf{w}}$
$\mathcal{Z}, \mathcal{W}$	Sets of network zones and workflows
$v_{i,t}^{(\mathrm{I})}, v_{i,t}^{(\mathrm{Z})}$	Intrusion state and zone of $i \in \mathcal{V}$ at time t
$v_{i,t}^{(\mathrm{R})}$	Reconnaissance state of $i \in \mathcal{V}$ at time t
$\Gamma, \mathcal{N}$	PO-POSG (7), set of players in Γ
$\mathbf{s}_t = (\mathbf{s}_t^{(\mathrm{D})}, \mathbf{s}_t^{(\mathrm{A})})$	State at time t
$\mathbf{a}_t = (\mathbf{a}_t^{(\mathrm{D})}, \mathbf{a}_t^{(\mathrm{A})})$	Action at time t
$\mathbf{o}_t, \mathbf{u}_t$	Observation and utility at time t
$\mathbf{a}_t^{(\mathrm{k})}, \mathbf{h}_t^{(\mathrm{k})}$	Action and history of player k at time t
$\mathcal{B}_{\mathrm{k}}, \mathbf{b}_t^{(\mathrm{k})}$	The belief space and belief state of player k
$\mathbf{S}_t, \mathbf{O}_t, \mathbf{A}_t$	Random vectors with realizations $\mathbf{s}_t, \mathbf{o}_t, \mathbf{a}_t$
$\mathbf{U}_t, \mathbf{B}_t^{(\mathrm{k})}, \mathbf{H}_t^{(\mathrm{k})}$	Random vectors with realizations $\mathbf{u}_t, \mathbf{b}_t^{(\mathrm{k})}, \mathbf{h}_t^{(\mathrm{k})}$
$u_{j,t}^{(\mathrm{w})}$	Workflow utility of node j at time t
π_{k}, Z	Strategy of player k and distribution of $\mathbf{o}_t$
$\tilde{\pi}_{\mathrm{k}}, \tilde{\mathbf{a}}^{(\mathrm{k})}$	Best response strategy and action of player k at time t
$\perp, \mathcal{K}_i$	The null action and nodes in the subtree of node i
$l_{i,t}$	Number of active nodes in $\mathcal{K}_i$ at time t
$\mathcal{A}_{\mathrm{D}}, \mathcal{A}_{\mathrm{A}}(\mathbf{s}_t)$	Action spaces at time t
$\mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}, \mathcal{A}_{\mathrm{A}}^{(\mathrm{V})}(\mathbf{s}_t)$	Node-local action spaces at time t, $\mathcal{A}_{\mathrm{k}} = (\mathcal{A}_{\mathrm{k}}^{(\mathrm{V})})^{|\mathcal{V}|}$
$f_{\mathrm{A}}, f_{\mathrm{D}}$	Attacker and defender transition functions
B_{k}	Best response correspondence of player k
u	Utility function of Γ
$\mathcal{S}, \mathcal{O}$	State and observation spaces of Γ
$\mathcal{O}^{(\mathrm{V})}$	Node-local observation space, $\mathcal{O} = (\mathcal{O}^{(\mathrm{V})})^{|\mathcal{V}|}$

TABLE 1: Notations for our mathematical model.

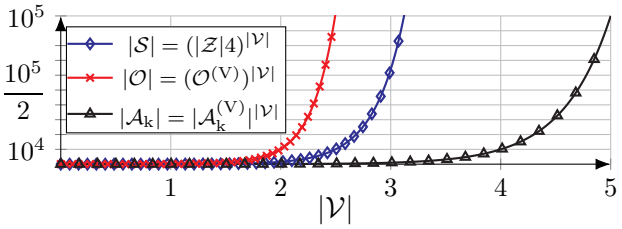
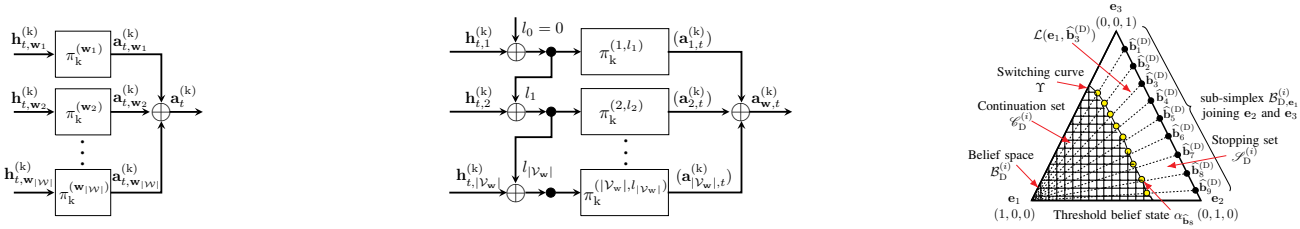


Fig. 6: Growth of $|\mathcal{S}|$, $|\mathcal{O}|$, and $|\mathcal{A}_{\mathrm{k}}|$ in function of the number of nodes $|\mathcal{V}|$, where $\mathrm{k} \in \{\mathrm{D}, \mathrm{A}\}$; the curves are computed using $|\mathcal{Z}| = 10$, $|\mathcal{O}^{(\mathrm{V})}| = 100$, and $|\mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}| = |\mathcal{A}_{\mathrm{A}}^{(\mathrm{V})}| = 10$.

Statements A and B express that Γ decomposes into $O(|\mathcal{V}||\mathcal{E}|)$ simpler subgames, which consequently can be solved in parallel (see Fig. 7). This decomposition implies that the largest game that is tractable on a given compute platform scales linearly with the number of processors. Further, statement C says that a best response strategy for the defender in each subgame can be characterized by a switching curve, which can be estimated efficiently.

In the following sections we provide proofs of Thm. 2.A–C. The requisite notations are given in Table 1.



(a) Theorem 2.A (b) Theorem 2.B (c) Theorem 2.C

Fig. 7: Illustrations of Thm. 2; arrows indicate inputs and outputs; $\oplus$ denotes vector concatenation; (a) illustrates that a game strategy π_{k} for player $\mathrm{k} \in \{\mathrm{D}, \mathrm{A}\}$ decomposes into $|\mathcal{W}|$ independent substrategies; (b) illustrates that a workflow strategy $\pi_{\mathrm{k}}^{(\mathbf{w})}$ for $\mathbf{w} \in \mathcal{W}$ decomposes into substrategies $(\pi_{\mathrm{k}}^{(i,l_i)})_{i \in \mathcal{V}_{\mathbf{w}}, l_i \in 1,\dots,|\mathcal{K}_i|}$ with optimal substructure; (c) provides a geometric illustration of the proof of Thm. 2.C, showing a switching curve that partitions the defender's belief space of a node $i \in \mathcal{V}$.

A. *Proof of Theorem 2.A*

Following the instantiation of Γ described in §IV, the state, observation, and action spaces factorize as

$$\mathcal{S} = (\mathcal{Z} \times \{0,1\}^2)^{|\mathcal{V}|} \quad \mathcal{O} = (\mathcal{O}^{(\mathrm{V})})^{|\mathcal{V}|} \quad \mathcal{A}_{\mathrm{k}} = (\mathcal{A}_{\mathrm{k}}^{(\mathrm{V})})^{|\mathcal{V}|} \tag{11}$$

where $\mathcal{O}^{(\mathrm{V})}$ and $\mathcal{A}^{(\mathrm{V})}$ denote the local observation and action spaces for each node, and $\mathrm{k} \in \{\mathrm{D}, \mathrm{A}\}$ denotes the player.

Since each node belongs to exactly one workflow, (11) implies that Γ can be decomposed into subgames $\Gamma^{(\mathbf{w}_1)}, \dots, \Gamma^{(\mathbf{w}_{|\mathcal{W}|})}$. To show that the subgames are independent, it suffices to show that the workflows are transition-independent and utility-independent [39, Defs. 2–3].

From the definitions of f_{D} and f_{A} in §IV we have

$$f_{\mathrm{D}}(\mathbf{S}_{i,t+1}^{(\mathrm{D})} | \mathbf{S}_t^{(\mathrm{D})}, \mathbf{A}_t^{(\mathrm{D})}) = f_{\mathrm{D}}(\mathbf{S}_{i,t+1}^{(\mathrm{D})} | \mathbf{S}_{i,t}^{(\mathrm{D})}, \mathbf{A}_{i,t}^{(\mathrm{D})})$$
$$f_{\mathrm{A}}(\mathbf{S}_{i,t+1}^{(\mathrm{A})} | \mathbf{S}_t^{(\mathrm{A})}, \mathbf{A}_t^{(\mathrm{A})}, \mathbf{A}_t^{(\mathrm{D})}) = f_{\mathrm{A}}(\mathbf{S}_{i,t+1}^{(\mathrm{A})} | \mathbf{S}_{i,t}^{(\mathrm{A})}, \mathbf{A}_{i,t}^{(\mathrm{A})}, \mathbf{A}_{i,t}^{(\mathrm{D})})$$

which implies transition independence across nodes $i \in \mathcal{V}$ and therefore across workflows [39, Def. 2].

From (4) we have that there exists a function $u_{\mathbf{w}}$ such that

$$\begin{aligned} u(\mathbf{s}_t, \mathbf{a}_t^{(\mathrm{D})}) &= \sum_{\mathbf{w} \in \mathcal{W}} \sum_{i \in \mathcal{V}_{\mathbf{w}}} \eta u^{(\mathrm{w})}(\mathbf{a}_{i,t}^{(\mathrm{D})}, l_{i,t}) - v_{i,t}^{(\mathrm{I})} \\ &= \sum_{\mathbf{w} \in \mathcal{W}} u_{\mathbf{w}}\Big((\mathbf{s}_{i,t}, \mathbf{a}_{i,t}^{(\mathrm{D})})_{i \in \mathcal{V}_{\mathbf{w}}}\Big) = \sum_{\mathbf{w} \in \mathcal{W}} u_{\mathbf{w},t} \end{aligned} \tag{12}$$

The final expression in (12) is a sum of local utility functions, each of which depends only on the states and actions of one workflow. Hence, $\Gamma^{(\mathbf{w}_1)}, \dots, \Gamma^{(\mathbf{w}_{|\mathcal{W}|})}$ are utility independent [39, Def. 3]. □

B. *Proof of Theorem 2.B*

Our goal is to show that a workflow subgame $\Gamma^{(\mathbf{w})}$ decomposes into node-level subgames with optimal substructure. Following the description in §IV, we know that the nodes in a workflow are connected in a tree and that the utility generated by a node i depends on the number of active nodes in the subtree rooted at i. Taking into account this tree structure and the definition of the utility function, we decompose $\Gamma^{(\mathbf{w})}$ into node subgames $(\Gamma^{(i,l_i)})_{i \in \mathcal{V}_{\mathbf{w}}, l_i \in \{1,\dots,|\mathcal{K}_i|\}}$ where l_i stands for the number of active nodes in the subtree rooted at i. It follows from (11) that this decomposition is feasible and that the space complexity of a subgame is independent of $|\mathcal{V}|$. Further, we know from Thm. 2.A that the subgames are transition-independent. It suffices now to show that at any time t, the best response action in $\Gamma^{(\mathbf{w})}$ for any node i is also a best response in $\Gamma^{(i,l_{i,t})}$ and vice versa. In this derivation, for better readability, we omit the constants γ, η and use the shorthand notations $\mathbf{s}_{\mathbf{w}}^{(\mathrm{D})} \triangleq (\mathbf{s}_j^{(\mathrm{D})})_{j \in \mathcal{V}_{\mathbf{w}}}$, $\mathbf{b}_{\mathbf{w}}^{(\mathrm{D})} \triangleq (\mathbf{b}_j^{(\mathrm{D})})_{j \in \mathcal{V}_{\mathbf{w}}}$, $V \triangleq V_{\mathrm{D},\pi_{\mathrm{A}}}^*$, and $\tau \in \arg\min_{k > t} \mathbf{a}_k^{(\mathrm{D})} \neq \bot$.

From Bellman's optimality equation [53, Eq. 1] a best response action for node i at time t in $\Gamma^{(\mathbf{w})}$ is given by

$$\begin{aligned} &\arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[\mathbf{U}_t + V(\mathbf{S}_{t+1}^{(\mathrm{D})}, \mathbf{B}_{t+1}^{(\mathrm{D})}) \middle| \mathbf{s}_t^{(\mathrm{D})}, \mathbf{b}_t^{(\mathrm{D})}, a_i^{(\mathrm{D})} \right] \right] \\ &\overset{(a)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[V(\mathbf{S}_{t+1}^{(\mathrm{D})}, \mathbf{B}_{t+1}^{(\mathrm{D})}) \middle| \mathbf{s}_t^{(\mathrm{D})}, \mathbf{b}_t^{(\mathrm{D})}, a_i^{(\mathrm{D})} \right] \right] \\ &\overset{(b)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[\sum_{k=t+1}^{\infty} \sum_{j \in \mathcal{K}_i} \mathbf{U}_{j,k} \middle| \overbrace{\mathbf{s}_{\mathbf{w},t}^{(\mathrm{D})}, \mathbf{b}_{\mathbf{w},t}^{(\mathrm{D})}, a_i^{(\mathrm{D})}}^{\triangleq \kappa} \right] \right] \\ &\overset{(c)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[\sum_{k=t+1}^{\tau} \sum_{j \in \mathcal{K}_i} \mathbf{U}_{j,k} \middle| \kappa \right] \right] \\ &\overset{(d)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[\sum_{k=t+1}^{\tau} \sum_{j \in \mathcal{K}_i} u_{j,k}^{(\mathrm{w})} - V_{i,k}^{(\mathrm{I})} \middle| \kappa \right] \right] \\ &\overset{(e)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[\sum_{k=t+1}^{\tau} V_{i,k}^{(\mathrm{I})} \middle| \kappa \right] \right] \\ &\overset{(f)}{=} \arg\max_{a_i^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}} \left[u_{i,t}^{(\mathrm{w})} + \mathbb{E}_{\pi_{\mathrm{A}}} \left[\sum_{k=t+1}^{\tau} V_{i,k}^{(\mathrm{I})} \middle| \mathbf{s}_{i,t}^{(\mathrm{D})}, \mathbf{b}_{i,t}^{(\mathrm{D})}, a_i^{(\mathrm{D})} \right] \right] \end{aligned} \tag{13}$$

where π_{A} denotes the attacker strategy and $\mathbf{U}_t$ denotes the vector of utilities for all nodes at time t. (a) holds because $(\mathbf{U}_{j,t})_{j \in \mathcal{V} \setminus \{i\}}$ is independent of $a_i^{(\mathrm{D})}$; (b) follows from (4) and the definition of the value function V [31, Thm. 7.4.1]; (c) holds because any $\mathbf{a}_{i,t}^{(\mathrm{D})}$ except $\bot$ leads to $\mathbf{s}_{i,t+1}^{(\mathrm{A})} = (0,0)$; (d) is a consequence of (4); (e) follows because $\mathbf{a}_{i,t}^{(\mathrm{D})}$ is independent of $(l_{i,k})_{k>t}$ (see §IV); and (f) is true due to transition independence across nodes (Thm. 2.A).

The last expression in (13) shows that the best response for node i depends on the other nodes only through $u_{i,t}^{(\mathrm{w})}$, which is a function of $l_{i,t}$. This means that a best response action

for node i in $\Gamma^{(\mathbf{w})}$ is also a best response in the subgame $\Gamma^{(i,l_{i,t})}$. □

C. Proof of Theorem 2.C

The idea behind this proof is that the problem of selecting *which* defensive action to apply in a node subgame $\Gamma^{(i,l_i)}$ can be separated from the problem of deciding *when* to apply it. Through this separation, we can analyze the latter problem using optimal stopping theory. Applying a recent result by Krishnamurthy [31, Thm. 12.3.4], the optimal stopping strategy in $\Gamma^{(i,l_i)}$ can be characterized by a switching curve.

By the chain rule of probability, a strategy $\pi_{\mathrm{D}}^{(i,l_i)}$ for $\Gamma^{(i,l_i)}$ (Thm. 2.B) can be represented as

$$\begin{aligned}
&\pi_{\mathrm{D}}^{(i,l_i)}(\mathbf{a}_i^{(\mathrm{D},1)}, \mathbf{a}_i^{(\mathrm{D},2)} \mid \mathbf{h}_{i,t}^{(\mathrm{k})}) \\
&= \pi_{\mathrm{D}}^{(i,l_i)}(\mathbf{a}_i^{(\mathrm{D},1)} \mid \mathbf{h}_{i,t}^{(\mathrm{D})}) \pi_{\mathrm{D}}^{(i,l_i)}(\mathbf{a}_i^{(\mathrm{D},2)} \mid \mathbf{h}_{i,t}^{(\mathrm{D})}, \mathbf{a}_i^{(\mathrm{D},1)}) \\
&\overset{(a)}{=} \pi_{\mathrm{D}}^{(i,l_i)}(\mathbf{a}_i^{(\mathrm{D},1)} \mid \mathbf{s}_{i,t}^{(\mathrm{D})}) \pi_{\mathrm{D}}^{(i,l_i)}(\mathbf{a}_i^{(\mathrm{D},2)} \mid \mathbf{b}_{i,t}^{(\mathrm{D})}, \mathbf{s}_{i,t}^{(\mathrm{D})}, \mathbf{a}_i^{(\mathrm{D},1)})
\end{aligned} \tag{14}$$

where $\mathbf{a}_i^{(\mathrm{D},1)} \neq \bot$ determines the action and $\mathbf{a}_i^{(\mathrm{D},2)} \in \{\mathrm{S}, \mathrm{C}\}$ determines when to take it. Specifically, if $\mathbf{a}_{i,t}^{(\mathrm{D},2)} = \mathrm{C}$, then $\mathbf{a}_{i,t}^{(\mathrm{D})} = \bot$, otherwise $\mathbf{a}_{i,t}^{(\mathrm{D})} = \mathbf{a}_{i,t}^{(\mathrm{D},1)}$. (a) follows because $\mathbf{a}_i^{(\mathrm{D})} \neq \bot$ causes $\mathbf{s}_i^{(\mathrm{A})} = (0,0)$ and is thus independent of $\mathbf{b}_i^{(\mathrm{D})}$. (Recall that the first component of $\mathbf{s}_i^{(\mathrm{A})}$ is the reconnaissance state and the second is the intrusion state.)

The action decomposition in (14) means that we can obtain a best response strategy in $\Gamma^{(i,l_i)}$ by jointly optimizing two substrategies: $\pi_{\mathrm{D}}^{(i,l_i,1)}$ and $\pi_{\mathrm{D}}^{(i,l_i,2)}$. The former corresponds to solving an MDP $\mathscr{M}^{(\mathrm{D},1)}$ with state space $\mathbf{s}^{(\mathrm{D})} \in \mathcal{Z}$ and the latter corresponds to solving a set of optimal stopping POMDPs $(\mathscr{M}^{(\mathrm{D},2)}_{i,s^{(\mathrm{D})},a^{(\mathrm{D})}})_{s^{(\mathrm{D})} \in \mathcal{Z}, a^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}$ with state space $\mathbf{s}^{(\mathrm{A})} \in \{(0,0),(1,0),(1,1)\}$. The belief space $\mathcal{B}_{\mathrm{D}}^{(i)}$ for each stopping problem is the 2-dimensional unit simplex.

The stopping problems are independent of time t because

$$\begin{aligned}
&\underset{\pi_{\mathrm{D}} \in \Pi_{\mathrm{D}}^{(i,l_i,2)}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{D}}} \left[\sum_{t=1}^{\infty} \gamma^{t-1} \mathbf{U}_{i,2,t} \;\middle|\; \mathbf{B}_{1,i}^{(\mathrm{D})} = \mathbf{e}_1 \right] \right] \\
&= \underset{\pi_{\mathrm{D}} \in \Pi_{\mathrm{D}}^{(i,2)}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{D}}} \left[\sum_{t=1}^{\tau_1} \gamma^{t-1} \mathbf{U}_{i,2,t} \;\middle|\; \mathbf{B}_{1,i}^{(\mathrm{D})} = \mathbf{e}_1 \right] + \right. \\
&\qquad \left. \mathbb{E}_{\pi_{\mathrm{D}}} \left[\sum_{t=\tau_1+1}^{\tau_2} \gamma^{t-1} \mathbf{U}_{i,2,t} \;\middle|\; \mathbf{B}_{\tau_1+1,i}^{(\mathrm{D})} = \mathbf{e}_1 \right] + \ldots \right] \\
&= \underset{\pi_{\mathrm{D}} \in \Pi_{\mathrm{D}}^{(i,2)}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{D}}} \left[\sum_{t=1}^{\tau_1} \gamma^{t-1} \mathbf{U}_{i,2,t} \;\middle|\; \mathbf{B}_{1,i}^{(\mathrm{D})} = \mathbf{e}_1 \right] \right]
\end{aligned} \tag{15}$$

where $\Pi_{\mathrm{D}}^{(i,2)}$, $\mathbf{U}_{i,2,t}$, and $\tau_1, \tau_2, \ldots$ denote the strategy space, utility, and stopping times in $\mathscr{M}^{(\mathrm{D},2)}_{i,s^{(\mathrm{D})},a^{(\mathrm{D})}}$. Note that $\mathbf{B}_{\tau+1,i}^{(\mathrm{D})} = \mathbf{e}_1 = (1,0,0)$ in (15) because each stop causes $\mathbf{s}_i^{(\mathrm{A})} = (0,0)$.

The transition probabilities for each stopping problem are given by the transition matrices:

$$\begin{bmatrix} 1-p & p & 0 \\ 0 & 1-q & q \\ 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix} \tag{16}$$

where p is the probability that the attacker peforms reconnaissance and q is the probability that the attacker compromises the node. The left matrix in (16) relates to $\mathbf{a}_{i,t}^{(\mathrm{D})} = \mathrm{C}$ and the right matrix relates to $\mathbf{a}_{i,t}^{(\mathrm{A})} = \mathrm{S}$. The non-zero second order minors of the matrices are $(1-p)(1-q)$, pq, $1-q$, $1-p$, and $(1-p)q$, which implies that the matrices are TP-2 [31, Def. 10.2.1]. Since the distributions $Z_{\mathbf{O}_1|\mathbf{s}^{(\mathrm{A})}}, \ldots, Z_{\mathbf{O}_{|\mathcal{V}|}|\mathbf{s}^{(\mathrm{A})}}$ also are TP-2 by assumption, it follows from [31, Thm. 12.3.4] that there exists a switching curve Υ that partitions $\mathcal{B}_{\mathrm{D}}^{(i)}$ into two individually connected regions: $\mathscr{S}_{\mathrm{D}}^{(i)}$ (the stopping set) and $\mathscr{C}_{\mathrm{D}}^{(i)}$ (the continuation set) (see Fig. 7c).

The main idea behind the proof of [31, Thm. 12.3.4] is that on any line segment $\mathcal{L}(\mathbf{e}_1, \widehat{\mathbf{b}^{(\mathrm{D})}})$ in $\mathcal{B}_{\mathrm{D}}^{(i)}$ that starts at $\mathbf{e}_1$ and ends at the subsimplex joining $\mathbf{e}_2$ and $\mathbf{e}_3$ (denoted with $\widehat{\mathbf{b}^{(\mathrm{D})}} \in \mathcal{B}_{\mathrm{D},\mathbf{e}_1}^{(i)}$), all belief states are totally ordered according to the Monotone Likelihood Ratio (MLR) order [31, Def. 10.1.1]. As a consequence, Topkis's theorem [54, Thm. 6.3] implies that the optimal strategy on $\mathcal{L}(\mathbf{e}_1, \widehat{\mathbf{b}^{(\mathrm{D})}})$ is monotone with respect to the MLR order. As a result, there exists a threshold belief state $\alpha_{\widehat{\mathbf{b}^{(\mathrm{D})}}}$ on $\mathcal{L}(\mathbf{e}_1, \widehat{\mathbf{b}^{(\mathrm{D})}})$ where the optimal strategy switches from C to S. Since $\mathcal{B}_{\mathrm{D}}^{(i)}$ can be covered by the union of lines $\mathcal{L}(\mathbf{e}_1, \widehat{\mathbf{b}^{(\mathrm{D})}})$, the thresholds $\alpha_{\widehat{\mathbf{b}^{(\mathrm{D})}}_1}, \alpha_{\widehat{\mathbf{b}^{(\mathrm{D})}}_2}, \ldots$ yield a switching curve Υ (see Fig. 7c). □

VII. Finding Nash Equilibria of the Decomposed Intrusion Response Game

To solve Γ (7) and find a Nash equilibrium we use a *fictitious self-play* algorithm called Decompositional Fictitious Self-Play (DFSP), which efficiently estimates a Nash equilibrium of Γ based on the decomposition presented above. The pseudocode of DFSP is listed in Alg. 1. (In Alg. 1, $\oplus$ denotes vector concatenation, $-\mathrm{k}$ denotes the opponent of player k, and $\mathscr{M}_{i,l_i}^{(\mathrm{k})}$ denotes the best response POMDP of k in $\Gamma^{(i,l_i)}$.)

DFSP implements the fictitious play process described in [55] and generates a sequence of strategy profiles $(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})$, $(\pi'_{\mathrm{D}}, \pi'_{\mathrm{A}})$, ... that converges to a Nash equilibrium $(\pi^*_{\mathrm{D}}, \pi^*_{\mathrm{A}})$ [56, Thms. 7.2.4–7.2.5]. During each step of this process, DFSP learns best responses against the players' current strategies and then updates both players' strategies (lines 8–12 in Alg. 1). To obtain the best responses, it first finds best responses for the node subgames induced by Thm. 2.B (lines 15–20) and then it combines them using the method described in §VI-B (lines 21–34).

Finding best responses for node subgames amounts to solving POMDPs. The principal method for solving POMDPs is dynamic programming [31], [57]. Dynamic programming is however intractable in our case, as demonstrated in Fig. 9b. To find the best responses we instead resort to approximation algorithms. More specifically, we use the Proximal Policy Optimization (PPO) algorithm [58] to find best responses for the attacker and we use a combination of dynamic programming and stochastic approximation to find best responses for the defender. In particular, to find best responses for the defender, we first solve the MDP defined in §VI-C via the value iteration algorithm [51], which can be done efficiently due to full observability. After solving the MDP, we approximate the

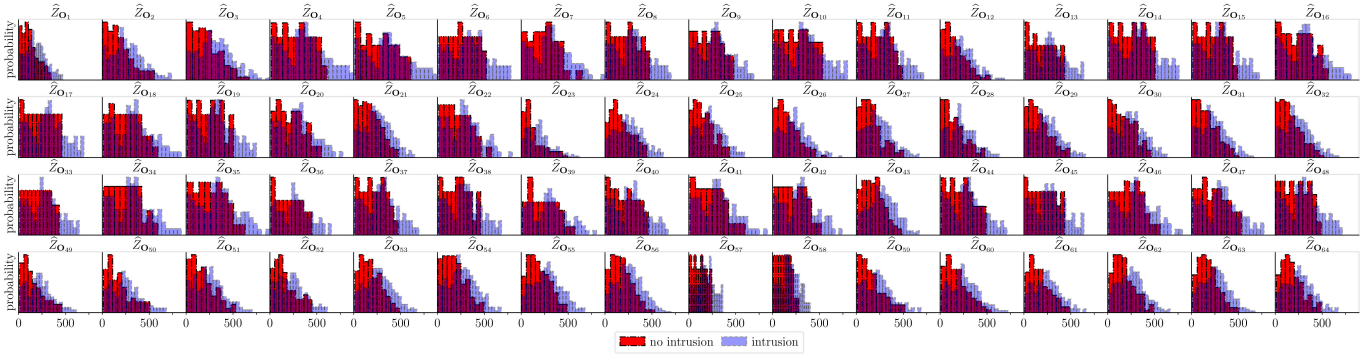


Fig. 8: Empirical observation distributions $\widehat{Z}_{\mathbf{O}_1}, \ldots, \widehat{Z}_{\mathbf{O}_{|\mathcal{V}|}}$ as estimates of $Z_{\mathbf{O}_1}, \ldots, Z_{\mathbf{O}_{|\mathcal{V}|}}$ in the target infrastructure (depicted in Fig. 2); $\mathbf{O}_i$ is a random variable representing the number of IDPS alerts related to node $i \in \mathcal{V}$, weighted by priority; the x-axes show the local observation spaces $\mathcal{O}^{(\mathrm{V})}$ for each node; the y-axes show $Z(\mathbf{O}_i \mid \mathbf{S}_i)$ (3).

Algorithm 1: DFSP: Decompositional Fictitious Self-Play

Input: P-SOLVER: a POMDP solver,
  δ: convergence criterion, Γ: the PO-POSG
Output: $(\pi^*_{\mathrm{D}}, \pi^*_{\mathrm{A}})$: an approximate Nash equilibrium
Algorithm DFSP (P-SOLVER, δ, Γ)
  Initialize $\pi_{\mathrm{D}}, \pi_{\mathrm{A}}, \widehat{\delta}$
  while $\widehat{\delta} \geq \delta$ **do**
    in parallel for $\mathrm{k} \in \{\mathrm{D}, \mathrm{A}\}$ **do**
      $\boldsymbol{\pi}_{\mathrm{k}} \leftarrow$ LOCAL-BRS(P-SOLVER, $\Gamma, \mathrm{k}, \pi_{-\mathrm{k}}$)
      $\tilde{\pi}_{\mathrm{k}} \leftarrow$ COMPOSITE-STRATEGY($\Gamma, \boldsymbol{\pi}_{\mathrm{k}}$)
      $\pi_{\mathrm{k}} \leftarrow$ AVERAGE-STRATEGY($\pi_{\mathrm{k}}, \tilde{\pi}_{\mathrm{k}}$)
    $\widehat{\delta} \leftarrow$ EXPLOITABILITY($\tilde{\pi}_{\mathrm{D}}, \tilde{\pi}_{\mathrm{A}}$)
  end
  return $(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})$
Procedure LOCAL-BRS(P-SOLVER, $\Gamma, \mathrm{k}, \pi_{-\mathrm{k}}$)
  $\boldsymbol{\pi}_{\mathrm{k}} \leftarrow ()$
  in parallel for $\mathbf{w} \in \mathcal{W}$ *and* $(i, l_i) \in \mathcal{V}_{\mathbf{w}} \times \mathcal{K}_i$ **do**
    $\boldsymbol{\pi}_{\mathrm{k}} \leftarrow \boldsymbol{\pi}_{\mathrm{k}} \oplus$ P-SOLVER($\mathscr{M}^{(\mathrm{k})}_{i,l_i}, \pi_{-\mathrm{k}}$)
  return $\boldsymbol{\pi}_{\mathrm{k}}$
Procedure COMPOSITE-STRATEGY($\Gamma, \boldsymbol{\pi}_{\mathrm{k}}$)
  $\pi_{\mathrm{k}} \leftarrow$ **Procedure** λ $(\mathbf{s}^{(\mathrm{k})}_t, \mathbf{b}^{(\mathrm{k})}_t)$
    $\mathbf{a}^{(\mathrm{k})}_t \leftarrow ()$
    for $\mathbf{w} \in \mathcal{W}$ *and* $i \in \mathcal{V}_{\mathbf{w}}$ **do**
      $\mathbf{a}^{(\mathrm{k})}_t \leftarrow \mathbf{a}^{(\mathrm{k})}_t \oplus (\boldsymbol{\pi}^{(i,l_{i,t})}_{\mathrm{k}}(\mathbf{s}^{(\mathrm{k})}_{i,t}, \mathbf{b}^{(\mathrm{k})}_{i,t}))$
    end
    return $\mathbf{a}^{(\mathrm{k})}_t$
  return π_{k}

optimal switching curves defined in §VI-C and Thm. 2.C with the following linear approximation [31, Eq. 12.18].

$$\pi_{\mathrm{D}}(\mathbf{b}^{(\mathrm{D})}) = \begin{cases} \text{Stop} & \text{if } \begin{bmatrix} 0 & 1 & \boldsymbol{\theta} \end{bmatrix} \begin{bmatrix} (\mathbf{b}^{(\mathrm{D})})^T \\ -1 \end{bmatrix} > 0 \\ \text{Continue} & \text{otherwise} \end{cases}$$
$$\text{subject to } \boldsymbol{\theta} \in \mathbb{R}^2,\ \boldsymbol{\theta}_2 > 0 \text{ and } \boldsymbol{\theta}_1 \geq 1 \qquad (17)$$

The coefficients $\boldsymbol{\theta}$ in (17) are estimated through the stochastic approximation algorithm in [31, Alg. 14] and [16, Alg. 1].

VIII. Digital Twin and System Identification

The DFSP algorithm described above approximates a Nash equilibrium of Γ (7) by simulating games and updating both players' strategies through reinforcement learning and dynamic programming. To identify the parameters required to instantiate these simulations and to evaluate the learned strategies, we use a digital twin of the target infrastructure (see Fig. 1). This section describes the digital twin (§VIII-A) and the identification process (§VIII-B).

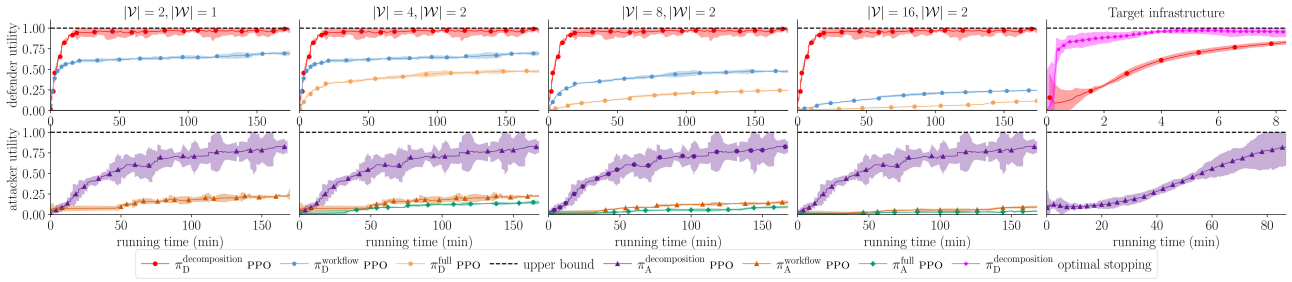
A. Creating a Digital Twin of the Target Infrastructure

The process of creating the digital twin involves two main tasks. The first task is to replicate relevant parts of the target infrastructure, such as physical resources, network interfaces, and network conditions. The second task is to emulate actors in the digital twin (i.e. the attacker, the defender, and the client population). Our implementation of the first task is based on network and resource virtualization and our implementation of the second task is based on traffic generators and automated attacks. Detailed descriptions of these implementations can be found in App. C and in our previous work [18].

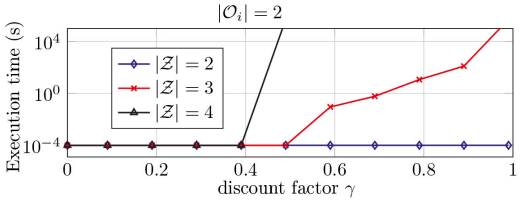
B. Estimating the Observation Distributions

Following the intrusion response use case described in §III, we define the observation $\mathbf{O}_{i,t}$ to be the number of IDPS alerts associated with node i at time t, weighted by priority. As our target infrastructure consists of 64 nodes (see App. C and Fig. 2), there are 64 alert distributions $Z_{\mathbf{O}_1}, \ldots, Z_{\mathbf{O}_{64}}$ (3). We estimate these distributions based on empirical data from the digital twin.

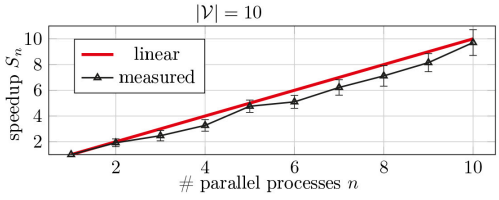
At the end of every time-step in the digital twin we collect the number of IDPS alerts that occurred during the time-step. These values are then used to compute the vector $\mathbf{o}_t$, which contains the total number of IDPS alerts per node, weighted by priority. For the evaluation in this paper we collect measurements from 10^4 time-steps using the Snort IDPS [59]. (Each time-step in the digital twin is 30 seconds.) Based on these measurements, we compute the empirical distributions $\widehat{Z}_{\mathbf{O}_1}, \ldots, \widehat{Z}_{\mathbf{O}_{64}}$ as estimates of $Z_{\mathbf{O}_1}, \ldots, Z_{\mathbf{O}_{64}}$ (see Fig. 8).



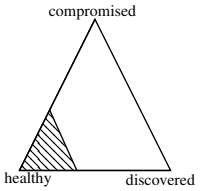
(a) Best response learning curves for the target infrastructure and synthetic infrastructures with varying $|\mathcal{V}|$ and $|\mathcal{W}|$.



(b) Runtimes of dynamic programming.



(c) Best response scalability.



(d) Best response structure.

Fig. 9: Best response learning via decomposition; (a) shows learning curves in simulation for synthetic infrastructures and the target infrastructure; the curves show the mean and 95% confidence interval for five random seeds; (b) shows execution times of computing best responses via dynamic programming and Sondik's value iteration algorithm [57]; (c) shows the speedup of our approach when computing best responses with different number of parallel processes; the speedup is calculated as $S_n = \frac{T_1}{T_n}$ where T_n is the completion time with n processes; and (d) shows an estimated switching curve (Thm. 2.C).

We observe in Fig. 8 that the distributions differ between nodes, which can be explained by the different services provided by the nodes (see App. C). We further observe that both the distributions when no intrusion occurs and the distributions during intrusion have most of their probability masses within $[0, 300]$. The distributions during intrusion also have substantial probability mass at larger values.

Remark: the stochastic matrices with the rows $\widehat{Z}_{\mathbf{O}_i|\mathbf{s}_i^{(\mathrm{A})}=(0,0)}$ and $\widehat{Z}_{\mathbf{O}_i|\mathbf{s}_i^{(\mathrm{A})}\neq(0,0)}$ have 250×10^9 second-order minors, which are almost all non-negative. This suggests that the TP-2 assumption in Thm. 2.C can be made.

IX. Experimental Evaluation

Our approach to find near-optimal defender strategies includes learning Nash equilibrium strategies via the DFSP algorithm and evaluating strategies in the digital twin (see Fig. 1). This section describes the evaluation results.

Experiment setup. The instantiation of Γ (7) and the hyperparameters are listed in App. A. We evaluate DFSP both for the target infrastructure (see Fig. 2 and App. C) and on synthetic infrastructures. The digital twin of the target infrastructure is deployed on a server with a 24-core INTEL XEON GOLD 2.10 GHz CPU and 768 GB RAM. Simulations of Γ and executions of DFSP run on a cluster with 2xTESLA P100 GPUS, 4xRTX8000 GPUS, and 3x16-core INTEL XEON 3.50 GHz CPU. Code and documentation for replicating the experiments are available in [18].

Convergence metric. To estimate the convergence of the sequence of strategy pairs generated by DFSP, we use the *approximate exploitability* metric $\widehat{\delta}$ [60]:

$$\widehat{\delta} = J(\widehat{\pi}_{\mathrm{D}}, \pi_{\mathrm{A}}) - J(\pi_{\mathrm{D}}, \widehat{\pi}_{\mathrm{A}}) \tag{18}$$

where $\widehat{\pi}_{\mathrm{k}}$ denotes an approximate best response strategy for player k. (J is defined in (4).) The closer $\widehat{\delta}$ becomes to 0, the closer $(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})$ is to a Nash equilibrium.

Baseline algorithms. We compare the performance of our approach ($\pi^{\text{decomposition}}$) with two baselines: π^{full} and π^{workflow}. Baseline π^{full} solves the full game without decomposition and π^{workflow} decomposes the game on the workflow-level only (appealing to Thm. 2.A).

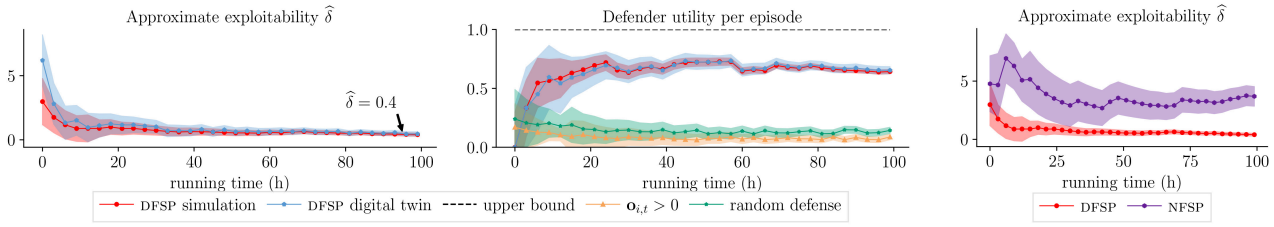
We compare the performance of DFSP with that of Neural Fictitious Self-Play (NFSP) [61] and PPO, which are the most popular algorithms among related work [12], [13], [62], [63].

Baseline strategies. We compare the strategies learned through DFSP with two baselines. The first baseline selects actions uniformly at random and the second baseline assumes prior knowledge of the opponent's actions and acts optimally based on this formation. We also compare the defender strategies against a heuristic baseline that shuts down a node $i \in \mathcal{V}$ as soon as an IDPS alert occurs, i.e. when $\mathbf{o}_{i,t} > 0$.

A. Learning Best Responses Against Static Opponents

We first examine whether our method can discover effective strategies against a *static* opponent strategy, which in game-theoretic terms is the problem of finding best responses (8)–(9). The static strategies are defined in App. B.

To measure the scalability of $\pi^{\text{decomposition}}$ we compare its performance with π^{workflow} and π^{full} on synthetic infrastructures with varying number of nodes $|\mathcal{V}|$ and workflows $|\mathcal{W}|$. Further, to evaluate the optimal stopping approach described in §VII we compare its rate of convergence in finding a best response for the target infrastructure with that of PPO. Figure 9a shows the learning curves. The red, purple, and pink curves represent the results obtained with $\pi^{\text{decomposition}}$; the blue and



(a) Nash equilibrium learning curves.

(b) Comparison DFSP and NFSP.

Fig. 10: Equilibrium learning via DFSP; the red curves show simulation results and the blue curves show emulation results; the green, orange, purple, and black curves relate to baselines; the figures show approximate exploitability (18) and normalized utility; the curves indicate the mean and the shaded areas indicate the standard deviation over three random seeds.

beige curves represent the results obtained with π^{workflow}; the orange and green curves represent the results obtained with π^{full}; and the dashed black lines relate to the baseline strategy that assumes prior knowledge of the opponent's strategy.

We note that all the learning curves of $\pi^{\text{decomposition}}$ converge, which suggests that the learned strategies converge as well. In contrast, the learning curves of π^{workflow} and π^{full} do not converge within the measured time. This is expected as π^{workflow} and π^{full} can not be parallelized like $\pi^{\text{decomposition}}$. (The speedup of parallelization is shown in Fig. 9c.) We further note that the converged curves of $\pi^{\text{decomposition}}$ are close to the dashed black lines, which indicates that the converged strategies are close to best responses. Lastly, we note in the rightmost plot of Fig. 9a that the optimal stopping approach, which exploits the statement in Thm. 2.C, converges significantly faster than PPO. An example of a learned optimal stopping strategy is shown in Fig. 9d.

B. Learning Equilibrium Strategies through Self-Play

Figure 10 shows the learning curves of the strategies obtained during the DFSP self-play process and the baselines introduced above. The red curves represent the results from the simulator; the blue curves show the results from the digital twin; the green curve give the performance of the random baseline; the orange curve relate to the $\mathbf{o}_{i,t} > 0$ baseline; and the dashed black line gives the performance of the baseline strategy that assumes prior knowledge of the attacker actions.

We note that all learning curves in Fig. 10a converge, which suggests that the learned strategies converge as well. Specifically, we observe that the approximate exploitability (18) 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 digital twin. Further, we see from the middle plot that both baseline strategies show decreasing performance as the attacker updates its strategy. In contrast, the defender strategy learned through DFSP improves its performance over time. This shows the benefit of a game-theoretic approach where the defender strategy is optimized against a dynamic attacker.

The right plot of Fig. 10 compares DFSP with NFSP on the simulator. NFSP implements fictitious self-play and can thus be compared with DFSP with respect to approximate exploitability (18). We observe in the left plot that DFSP converges faster than NFSP. The fast convergence of DFSP in comparison with NFSP is expected as DFSP is parallelizable while NFSP is not.

C. Discussion of the Evaluation Results

In this work, we propose a framework based on recursive decomposition for 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. 10a). 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 digital twin as in the simulator gives us confidence that the obtained strategies would perform as expected in the target infrastructure.

(ii) Decomposition provides a scalable approach to automate intrusion responses for IT infrastructures (see Fig. 9a and Fig. 10b). The intuition behind this finding is that decomposition allows to design efficient "piece-by-piece" algorithms that can be parallelized (Thm. 2.A–B).

(iii) The theory of optimal stopping provides insight about optimal defender strategies, which enables efficient computation of best responses (see the rightmost plot in Fig. 9a). This finding can be explained by the threshold structures of the best response strategies, which drastically reduce the search space of possible strategies (Thm. 2.C).

(iv) Static defender strategies' performance deteriorate against a dynamic attacker whereas defender strategies obtained through DFSP improve over time (see the middle plot in Fig. 10). This finding is consistent with previous studies that use game-theoretic approaches (e.g. [41] and [17]) and suggests limitations of static intrusion response systems, such as the Snort IDPS.

X. Conclusions and Future Work

We combine reinforcement learning, decomposition, and a digital twin in a game-theoretic framework to address the problem of automated intrusion response, which we formulate as a partially observed stochastic game. We prove that the game decomposes recursively into subgames that can be solved in parallel and show that the optimal defender strategies exhibit threshold structures. This decomposition provides us with a

Game parameters	Values
$u_{\mathbf{w},t}$, $\mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}$	$\sum_{i\in\mathcal{V}_{\mathbf{w}}}[\mathrm{gw}\rightarrow_t i]$, $\mathcal{Z}\cup\{\text{access control},\perp\}$
$\lvert\mathcal{O}^{(\mathrm{V})}\rvert$, γ, η, $\lvert\mathcal{Z}\rvert$, $\lvert\mathcal{W}\rvert$, $\lvert\mathcal{V}\rvert$	10^3, 0.9, 0.4, 6, 10, 64
$u_i^{(\mathrm{w})}(\perp,l)$, $u_i^{(\mathrm{w})}(\mathfrak{S},l)$,$u_i^{(\mathrm{w})}(\mathfrak{R},l)$, $u_i^{(\mathrm{w})}(2,l)$	0, $10+l$, $15+l$, $0.1+l$
$u_i^{(\mathrm{w})}(3,l)$, $u_i^{(\mathrm{w})}(4,l)$, $u_i^{(\mathrm{w})}(5,l)$, $u_i^{(\mathrm{w})}(0.8,l)$	$0.5+l$, $1+l$, $1.5+l$, $2+l$
topology $\mathcal{G}$ and $\mathbf{s}_1^{(\mathrm{D})}$	see Fig. 2
$\lvert\mathcal{V}_{\mathbf{w}_1}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_2}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_3}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_4}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_5}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_6}\rvert$	16, 16, 16, 16, 6, 4
$\lvert\mathcal{V}_{\mathbf{w}_7}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_8}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_9}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_{10}}\rvert$	6, 4, 6, 6
PPO parameters	
lr α, batch, # layers, # neurons, clip ϵ	10^{-5}, $4\cdot10^3t$, 4, 64, 0.2
GAE λ, ent-coef, activation	0.95, 10^{-4}, ReLU
NFSP parameters	
lr RL, lr SL, batch, # layers,# neurons, $\mathcal{M}_{RL}$	10^{-2}, $5\cdot10^{-3}$, 64, 2,128, 2×10^5
$\mathcal{M}_{SL}$,ϵ, ϵ-decay, η	2×10^6, 0.06, 0.001, 0.1
Stochastic approximation parameters	
c, ϵ, λ, A, a, N, δ	10, 0.101, 0.602, 100, 1, 50, 0.2

TABLE 2: Hyperparameters ($[\cdot]$ is the Iverson bracket).

scalable approach to learn near-optimal defender strategies, based on which we develop Decompositional Fictitious Self-Play (DFSP) – a fictitious self-play algorithm for finding Nash equilibria. To assess the learned strategies in a real environment, we evaluate them in a digital twin of the target infrastructure. The results from running DFSP demonstrate that the learned strategies converge to an approximate Nash equilibrium and thus to near-optimal strategies both in simulation and in the digital twin. The results also demonstrate that DFSP converges faster than a state-of-the-art algorithm by taking advantage of the game decomposition. In future work we plan to generalize our framework to additional use cases.

XI. Acknowledgments

The authors would like to thank Pontus Johnson and Quanyan Zhu for useful inputs to this research. The authors are also grateful to Forough Shahab Samani and Xiaoxuan Wang for their constructive comments on a draft of this paper.

Appendix A
Hyperparameters and Game Instantiation

We instantiate Γ (7) for the experimental evaluation as follows. Client arrivals are sampled from a stationary Poisson process $Po(\lambda = 50)$ and service times are exponentially distributed with mean $\mu = 4$. In addition to migrate a node, the defender can shut it down or redirect its traffic to a honeynet, which we model with the zones $\mathfrak{S},\mathfrak{R}\in\mathcal{Z}$. A node $i\in\mathcal{V}$ is shutdown if $v_{i,t}^{(\mathrm{Z})}=\mathfrak{S}$ and have its traffic redirected if $v_{i,t}^{(\mathrm{Z})}=\mathfrak{R}$. The set of local attacker actions is $\mathcal{A}_{\mathrm{A}}^{(\mathrm{V})}=\{\perp,\text{reconnaissance},\text{brute-force},\text{exploit}\}$, which we encode as $\{0,1,2,3\}$. These actions have the following effects on the state $\mathbf{s}_t$: $\mathbf{a}_{i,t}^{(\mathrm{A})}=1\implies v_{i,t}^{(\mathrm{R})}=1$, $\mathbf{a}_{i,t}^{(\mathrm{A})}=2\implies v_{i,t}^{(\mathrm{I})}=1$ with probability 0.3, and $\mathbf{a}_{i,t}^{(\mathrm{A})}=3\implies v_{i,t}^{(\mathrm{I})}$ with probability 0.4. We enforce a tree structure on the target infrastructure in Fig. 2 by disregarding the redundant edges in the R&D zone. The remaining parameters are listed in Table 2.

Appendix B
Static Defender and Attacker Strategies

The static defender and attacker strategies for the evaluation described in §IX-A are defined in (19)–(20) and the actions to emulate the attacker are listed in Table 3. (w.p is short for "with probability".)

Type	*Actions*
Reconnaissance	TCP SYN scan, UDP port scan, TCP XMAS scan VULSCAN vulnerability scanner, ping-scan
Brute-force	TELNET, SSH, FTP, CASSANDRA, IRC, MONGODB, MYSQL, SMTP, POSTGRES
Exploit	CVE-2017-7494, CVE-2015-3306, CVE-2010-0426, CVE-2015-5602,CVE-2015-1427 CVE-2014-6271, CVE-2016-10033, CWE-89

TABLE 3: Attacker actions in the digital twin; exploits are identified according to the Common Vulnerabilities and Exposures (CVE) database [64] and the Common Weakness Enumeration (CWE) list [65].

$$\pi_{\mathrm{D}}(\mathbf{h}_t^{(\mathrm{D})})_i=\begin{cases}\perp & \text{w.p } 0.95\\ j\in\mathcal{Z} & \text{w.p } \dfrac{0.05}{|\mathcal{Z}|+1}\end{cases}\tag{19}$$

$$\pi_{\mathrm{A}}(\mathbf{h}_t^{(\mathrm{A})})_i=\begin{cases}\perp & \text{if } v_{i,t}^{(\mathrm{I})}=1\\ \perp & \text{w.p } 0.8 \text{ if } v_{i,t}^{(\mathrm{R})}=0\\ \perp & \text{w.p } 0.7 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0\\ \text{recon} & \text{w.p } 0.2 \text{ if } v_{i,t}^{(\mathrm{R})}=0\\ \text{brute} & \text{w.p } 0.15 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0\\ \text{exploit} & \text{w.p } 0.15 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0\end{cases}\tag{20}$$

Appendix C
Configuration of the Infrastructure in Fig. 2

The configuration of the target infrastructure (Fig. 2) is available in Tables 4 and 5.

References

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

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

[3] S. Moothedath, D. Sahabandu, J. Allen, A. Clark, L. Bushnell, W. Lee, and R. Poovendran, "A game-theoretic approach for dynamic information flow tracking to detect multistage advanced persistent threats," *IEEE Transactions on Automatic Control*, 2020.

[4] D. Umsonst, S. Sarıtaş, G. Dán, and H. Sandberg, "A bayesian nash equilibrium-based moving target defense against stealthy sensor attacks," 2022.

[5] E. Altman, K. Avrachenkov, and A. Garnaev, "Jamming game with incomplete information about the jammer," in *Conference on Performance Evaluation Methodologies and Tools*, 2009.

[6] E. E. Tsiropoulou, J. Baras, S. Papavassiliou, and G. Qu, "On the mitigation of interference imposed by intruders in passive rfid networks," 11 2016.

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

[8] 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.

ID(s)	*Type*	*Operating system*	*Zone*	*Services*	*Vulnerabilities*
1	Gateway	UBUNTU 20	-	SNORT (ruleset v2.9.17.1), SSH, OPENFLOW v1.3, RYU SDN controller	-
2	Gateway	UBUNTU 20	DMZ	SNORT (ruleset v2.9.17.1), SSH, OVS v2.16, OPENFLOW v1.3	-
28	Gateway	UBUNTU 20	R&D	SNORT (ruleset v2.9.17.1), SSH, OVS v2.16, OPENFLOW v1.3	-
3,12	Switch	UBUNTU 22	DMZ	SSH, OPENFLOW v1.3 , OVS v2.16	-
21, 22	Switch	UBUNTU 22	-	SSH, OPENFLOW v1.3, OVS v2.16	-
23	Switch	UBUNTU 22	ADMIN	SSH, OPENFLOW v1.3, OVS v2.16	-
29-48	Switch	UBUNTU 22	R&D	SSH, OPENFLOW v1.3, OVS v2.16	-
13-16	Honeypot	UBUNTU 20	DMZ	SSH, SNMP, POSTGRES, NTP	-
17-20	Honeypot	UBUNTU 20	DMZ	SSH, IRC, SNMP, SSH, POSTGRES	-
4	App server	UBUNTU 20	DMZ	HTTP, DNS, SSH	CWE-1391
5, 6	App server	UBUNTU 20	DMZ	SSH, SNMP, POSTGRES, NTP	-
7	App server	UBUNTU 20	DMZ	HTTP, TELNET, SSH	CWE-1391
8	App server	DEBIAN JESSIE	DMZ	FTP, SSH, APACHE 2,SNMP	CVE-2015-3306
9,10	App server	UBUNTU 20	DMZ	NTP, IRC, SNMP, SSH, POSTGRES	-
11	App server	DEBIAN JESSIE	DMZ	APACHE 2, SMTP, SSH	CVE-2016-10033
24	Admin system	UBUNTU 20	ADMIN	HTTP, DNS, SSH	CWE-1391
25	Admin system	UBUNTU 20	ADMIN	FTP, MONGODB, SMTP, TOMCAT, TS 3, SSH	-
26	Admin system	UBUNTU 20	ADMIN	SSH, SNMP, POSTGRES, NTP	-
27	Admin system	UBUNTU 20	ADMIN	FTP, MONGODB, SMTP, TOMCAT, TS 3, SSH	CWE-1391
49-59	Compute server	UBUNTU 20	R&D	SPARK, HDFS	-
60	Compute server	DEBIAN WHEEZY	R&D	SPARK, HDFS, APACHE 2,SNMP, SSH	CVE-2014-6271
61	Compute server	DEBIAN 9.2	R&D	IRC, APACHE 2, SSH	CWE-89
62	Compute server	DEBIAN JESSIE	R&D	SPARK, HDFS, TS 3, TOMCAT, SSH	CVE-2010-0426
63	Compute server	DEBIAN JESSIE	R&D	SSH, SPARK, HDFS	CVE-2015-5602
64	Compute server	DEBIAN JESSIE	R&D	SAMBA, NTP, SSH, SPARK, HDFS	CVE-2017-7494

TABLE 4: Configuration of the target infrastructure shown in Fig. 2; each row contains the configuration of one or more components; vulnerabilities are identified according to the CVE and CWE databases [64], [65].

ID	*Name*	*Zone*	*Components*
1	SPARK 1	R&D	1, 21, 22, 28, (29 − 32), (33 − 34), (41 − 42), (49 − 52)
2	SPARK 2	R&D	1, 21, 22, 28, (29 − 32), (35 − 36), (43 − 44), (53 − 56)
3	SPARK 3	R&D	1, 21, 22, 28, (29 − 32), (37 − 38), (45 − 46), (57 − 60)
4	SPARK 4	R&D	1, 21, 22, 28, (29 − 32), (39 − 40), (47 − 48), (61 − 65)
5	Web 1	DMZ	1, 2, 3, 4, 5, 6
6	Web 2	DMZ	1, 2, 3, 7
7	Storage 1	DMZ	1, 2, 3, 8, 9, 10
8	Mail 1	DMZ	1, 2, 3, 11
9	Admin 1	ADMIN	1, 21, 22, 23, 24, 25
10	Admin 2	ADMIN	1, 21, 22, 23, 25, 26

TABLE 5: Workflows of the target infrastructure (Fig. 2).

[9] 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.

[10] 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.

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

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

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

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

[15] ——, "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.

[16] ——, "Intrusion prevention through optimal stopping," *IEEE Transactions on Network and Service Management*, vol. 19, no. 3, pp. 2333–2348, 2022.

[17] ——, "Learning near-optimal intrusion responses against dynamic attackers," 2023.

[18] K. Hammar, "Cyber security learning environment," 2023, https://limmen.dev/csle/.

[19] 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.

[20] 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.

[21] 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*, 2022.

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

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

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

[25] C. Kamhoua, C. Kiekintveld, F. Fang, and Q. Zhu, *Game Theory and Machine Learning for Cyber Security*. Wiley, 2021.

[26] S. Zonouz *et al.*, "Rre: A game-theoretic intrusion response and recovery engine," in *2009 IEEE/IFIP International Conference on Dependable Systems & Networks*, 2009.

[27] C. Shi, S. Han, and J. Fu, "Quantitative planning with action deception in concurrent stochastic games," 2023.

[28] O. Tsemogne, Y. Hayel, C. Kamhoua, and G. Deugoué, "Optimizing intrusion detection systems placement against network virus spreading using a partially observable stochastic minimum-threat path game," in *Decision and Game Theory for Security*, 2023, pp. 274–296.

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

[30] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 4th ed. The MIT Press, 2022.

[31] V. Krishnamurthy, *Partially Observed Markov Decision Processes: From Filtering to Controlled Sensing*, 2016.

[32] J. Anderson and A. Papachristodoulou, "Dynamical system decomposition for efficient, sparse analysis," in *49th IEEE Conference on Decision and Control (CDC)*, 2010, pp. 6565–6570.

[33] Y. Ouyang, H. Tavafoghi, and D. Teneketzis, "Dynamic games with asymmetric information: Common information based perfect bayesian equilibria and sequential decomposition," *IEEE Transactions on Automatic Control*, vol. 62, no. 1, pp. 222–237, 2017.

[34] D. Siljak, *Large-scale Dynamic Systems: Stability and Structure*, ser. North-Holland series in system science and engineering, 1978.

[35] J. D. Day and H. Zimmermann, "The OSI reference model," *Proceedings of the IEEE*, vol. 71, no. 12, pp. 1334–1340, 1983.
[36] R. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal on Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.
[37] M. Kearns and D. Koller, "Efficient reinforcement learning in factored mdps," in *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, ser. IJCAI'99, San Francisco, CA, USA, 1999.
[38] S. Singh and D. Cohn, "How to dynamically merge markov decision processes," in *Advances in Neural Information Processing Systems*, M. Jordan, M. Kearns, and S. Solla, Eds., vol. 10. MIT Press, 1997.
[39] R. Becker, S. Zilberstein, V. Lesser, and C. V. Goldman, "Transition-independent decentralized markov decision processes," in *Joint Conference on Autonomous Agents and Multiagent Systems*, 2003.
[40] R. Nair *et al.*, "Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps," in *Conference on Artificial Intelligence and the Innovative Applications of Artificial Intelligence*, 2005.
[41] L. Huang, J. Chen, and Q. Zhu, *Factored Markov Game Theory for Secure Interdependent Infrastructure Networks*, 2018.
[42] T. Kroupa, "On decomposition of solutions for coalitional games," 2020.
[43] 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.
[44] J. Zheng and D. A. Castañón, "Decomposition techniques for markov zero-sum games with nested information," in *52nd IEEE Conference on Decision and Control*, 2013, pp. 574–581.
[45] X. Zan *et al.*, "A hierarchical and factored pomdp based automated intrusion response framework," in *2010 2nd International Conference on Software Technology and Engineering*, 2010.
[46] J. von Neumann, "Zur Theorie der Gesellschaftsspiele. (German) [On the theory of games of strategy]," vol. 100, pp. 295–320, 1928.
[47] J. F. Nash, "Non-cooperative games," *Annals of Mathematics*, 1951.
[48] K. Horák and B. Bošanský, "Solving partially observable stochastic games with public observations," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, pp. 2029–2036, Jul. 2019.
[49] 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.
[50] K. Horák, "Scalable algorithms for solving stochastic games with limited partial observability," Ph.D. dissertation, 2019.
[51] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed., USA, 1994.
[52] S. Banach, "Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales," *Fundamenta Mathematicae*, 1922.
[53] R. Bellman, "A markovian decision process," *Journal of Mathematics and Mechanics*, vol. 6, no. 5, pp. 679–684, 1957.
[54] D. M. Topkis, "Minimizing a submodular function on a lattice," *Operations Research*, vol. 26, no. 2, pp. 305–321, 1978.
[55] G. W. Brown, "Iterative solution of games by fictitious play," 1951, activity analysis of production and allocation.
[56] Y. Shoham and K. Leyton-Brown, *Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*, Cambridge, UK, 2009.
[57] E. J. Sondik, "The optimal control of partially observable markov processes over the infinite horizon: Discounted costs," *Operations Research*, vol. 26, no. 2, pp. 282–304, 1978.
[58] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, 2017, http://arxiv.org/abs/1707.06347.
[59] 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.
[60] F. Timbers *et al.*, "Approximate exploitability: Learning a best response in large games," 2020, https://arxiv.org/abs/2004.09677.
[61] J. Heinrich and D. Silver, "Deep reinforcement learning from self-play in imperfect-information games," *CoRR*, vol. abs/1603.01121, 2016.
[62] 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.
[63] W. Xue *et al.*, "Solving large-scale extensive-form network security games via neural fictitious self-play," 2021.
[64] T. M. Corporation, "Cve database," 2022, https://cve.mitre.org/.
[65] ——, "Cwe list," 2023, https://cwe.mitre.org/index.html.