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 23, 2023

***Abstract*—We study automated intrusion response for an IT infrastructure and formulate the interaction between an attacker and a defender as a partially observed stochastic game. To solve the game we follow an approach where attack and defense strategies co-evolve through reinforcement learning and self-play toward an equilibrium. Solutions proposed in previous work prove the feasibility of this approach for small infrastructures but do not scale to realistic scenarios due to the exponential growth in computational complexity with the infrastructure size. We address this problem by introducing a method that recursively decomposes the game into subgames which can be solved in parallel. Applying optimal stopping theory we show that the best response strategies in these subgames exhibit threshold structures, which allows us to compute them efficiently. To solve the decomposed game we introduce an algorithm called Decompositional Fictitious Self-Play (DFSP), which learns Nash equilibria through stochastic approximation. We evaluate the learned strategies in an emulation environment where real intrusions and response actions can be executed. The results show that the learned strategies approximate an equilibrium and that DFSP significantly outperforms a state-of-the-art algorithm for a realistic infrastructure configuration.**

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

I. Introduction

A promising direction of recent research is to automatically find security strategies for an IT infrastructure through reinforcement learning methods, whereby the problem is formulated as a Markov decision problem and strategies are learned through simulation (see survey [1]). While encouraging results have been obtained following this approach (see e.g. [2] and [3]), key challenges remain. For example, most of the prior work follows a decision-theoretic formulation and aims at learning effective defender strategies against a static attacker with a fixed strategy [2]–[14]. Only recently has the problem of learning effective security strategies against dynamic attackers been studied. This approach includes a game-theoretic framing, and the problem becomes one of learning Nash equilibria [15]–[24].

Chief among the remaining challenges is the complexity of the formal model, resulting from the need to describe the target infrastructure with sufficient detail and at a realistic scale. Learning effective strategies with currently known methods is infeasible for most realistic use cases.

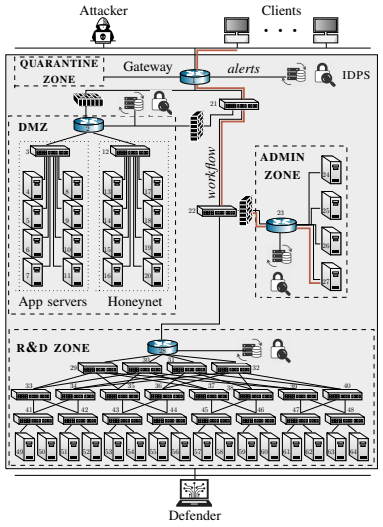


Fig. 1: The target infrastructure and the actors involved in the intrusion response use case.

In this paper, we address the complexity challenge and present a scalable approach to automatically learn near-optimal defender strategies against dynamic attackers. We apply our approach to an *intrusion response* use case that involves the IT infrastructure of an organization (see Fig. 1). We formalize the 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. To manage the complexity when formalizing the use case, we recursively decompose the game into simpler subgames, which allows detailed modeling of the infrastructure while keeping computational complexity low.

The decomposition involves three steps. First, we partition the infrastructure according to workflows that are isolated from each other. This allows us to decompose the game into *independent subgames* (one per workflow) that can be solved in parallel. Second, we exploit the fact that workflows usually have graph structure, which allows us to decompose the workflow games into node subgames. We prove that these subgames have *optimal substructure* [25, Ch. 15], which means that a best response of the original game can be obtained from best responses of the node subgames. Third, we show that the problem of selecting *which* response action to

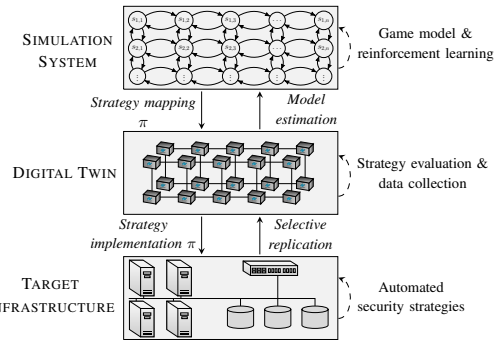


Fig. 2: Our framework for finding and evaluating intrusion response strategies [3], [6], [19].

apply on a node can be separated from that of deciding *when* to apply the action, which enables efficient learning of best responses through the application of *optimal stopping* theory [26]. We use this result to design an efficient reinforcement learning algorithm, called Decompositional Fictitious Self-Play (DFSP), which allows scalable approximation of Nash equilibrium strategies.

Our experimental method for learning the equilibrium strategies and evaluating them is based on a *digital twin* of the target infrastructure, which we use to run attack scenarios and defender responses (see Fig. 2) [3], [6], [19]. Such runs produce system measurements and logs, from which we estimate infrastructure statistics. We then use these statistics to instantiate simulations of the infrastructure's dynamics and learn strategies through DFSP.

We summarize the contributions in this paper as follows.

1) We formulate the intrusion response problem as a partially observed stochastic game and prove that, under assumptions often met in practice, the game decomposes into subgames whose best responses can be computed efficiently and in parallel.
2) We design DFSP, an efficient reinforcement learning algorithm for approximating Nash equilibria of the decomposed game.
3) For a realistic use case, we evaluate the learned response strategies against real network intrusions on a digital twin.

II. Related Work

Networked systems found in engineering and science often exhibit a modular topological structure that can be exploited for designing control algorithms [27]. System decomposition for the purpose of automatic control was first suggested by Šiljak in 1978 [28] and approaches based on decomposition, such as divide and conquer, layering, and hierarchical structuring are well established in the design of large-scale systems, a notable example being the Internet [29]. Similar decomposition methods are frequently used in robotics and multi-agent systems, as exemplified by the subsumption architecture [30]. Within the fields of decision- and game-theory, decomposition is studied in the context of factored decision processes [31]–[34], distributed decision processes [35], factored games [36], [37], and graphical games [38].

Decomposition as a means to automate intrusion responses has been studied first in [36], [39]–[41]. The work in [36] formulates the interaction between a defender and an attacker on a cyber-physical infrastructure as a factored Markov game and introduces a decomposition based on linear programming. Following a similar approach, the work in [40] studies a Markov game formulation and shows that a multi-stage game can be decomposed into a sequence of one-stage games. In a separate line of work, [39] 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 [41], which studies a factored decision process and proposes a hierarchical decomposition.

In all of the above works, decomposition is key to obtain effective strategies for large-scale systems. Compared to our work, some of them propose decomposition methods without optimal substructure [39], others do not consider partial observability [36], [40], or dynamic attackers [41]. Most importantly, all of the above works evaluate the obtained strategies in a simulation environment. They do not perform evaluation in an emulation environment as we report in this paper, which gives higher confidence that the strategies are effective on the target infrastructure.

For a comprehensive review of prior research on automated intrusion response (beyond work that use decomposition), see [19, §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. 1). 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 servers between zones; (*ii*) redirect or block network flows; (*iii*) shut down servers; and (*iv*) revoke access to servers (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 the intrusion response problem. The game is played on the IT infrastructure, which we model as a discrete-time dynamical

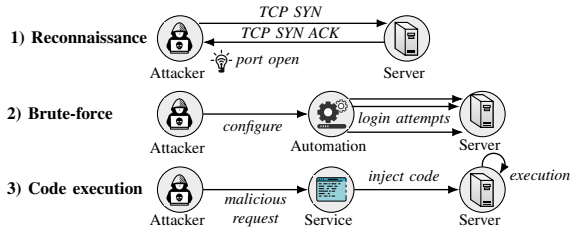


Fig. 3: Attacker actions: (*i*) reconnaissance actions; (*ii*) brute-force attacks; and (*iii*) code execution attacks.

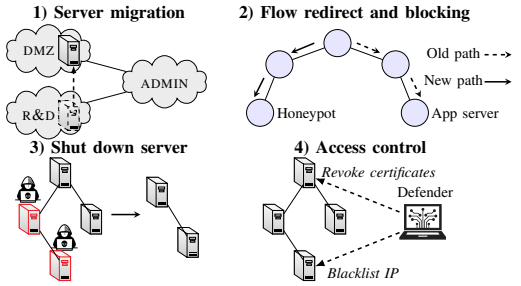


Fig. 4: Defender actions: (*i*) migrate a server between two zones; (*ii*) redirect or block traffic flows to a server; (*iii*) shut down a server; and (*iv*) revoke access to a server.

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 random 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. 1). 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 $\mathcal{G} \triangleq \langle \{\mathrm{gw}\} \cup \mathcal{V}, \mathcal{E} \rangle$. The graph has a tree structure and is rooted at the gateway gw. Each node $i \in \mathcal{V}$ has three state variables. $v_{i,t}^{(\mathrm{R})}$ represents the reconnaissance state and realizes the binary random variable $V_{i,t}^{(\mathrm{R})}$. $V_{i,t}^{(\mathrm{R})} = 1$ if the attacker has discovered the node, 0 otherwise. $v_{i,t}^{(\mathrm{I})}$ represents the intrusion state and realizes the binary random variable $V_{i,t}^{(\mathrm{I})}$. $V_{i,t}^{(\mathrm{I})} = 1$ if the attacker has compromised the node, 0 otherwise. Lastly, $v_{i,t}^{(\mathrm{Z})}$ indicates the zone in which the node resides and realizes the random variable $V_{i,t}^{(\mathrm{Z})}$. We call a node *active* if it is functional as part of a workflow (denoted $\alpha_{i,t} = 1$). Due to defender actions (e.g. shut downs) a node $i \in \mathcal{V}$ may become inactive (i.e. $\alpha_{i,t} = 0$).

Workflows. We model a workflow $\mathbf{w} \in \mathcal{W}$ as a subtree $\mathcal{G}_\mathbf{w} \triangleq \langle \{\mathrm{gw}\} \cup \mathcal{V}_\mathbf{w}, \mathcal{E}_\mathbf{w} \rangle$ 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: an attacker, a defender, and clients (see Fig. 1).

Attacker. At each time t, the attacker takes an action $\mathbf{a}_t^{(\mathrm{A})}$, which is defined as the composition of the local actions on all nodes $\mathbf{a}_t^{(\mathrm{A})} \triangleq (\mathbf{a}_{1,t}^{(\mathrm{A})}, \ldots, \mathbf{a}_{|\mathcal{V}|,t}^{(\mathrm{A})}) \in \mathcal{A}_\mathrm{A}$, where $\mathcal{A}_\mathrm{A}$ 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 $V_{i,t}^{(\mathrm{R})}$ or the intrusion state $V_{i,t}^{(\mathrm{I})}$. A node i can only be compromised if it is discovered, i.e. if $V_{i,t}^{(\mathrm{R})} = 1$. We denote this constraint with $\mathbf{a}_t^{(\mathrm{A})} \in \mathcal{A}_\mathrm{A}(\mathbf{s}_t)$.

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

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

where $\mathbf{S}_t^{(\mathrm{A})}$, $\mathbf{A}_t^{(\mathrm{A})}$, and $\mathbf{A}_t^{(\mathrm{D})}$ are random vectors with realizations $\mathbf{s}_t^{(\mathrm{A})}$, $\mathbf{a}_t^{(\mathrm{A})}$, and $\mathbf{a}_t^{(\mathrm{D})}$.

Defender. At each time t, the defender takes an action $\mathbf{a}_t^{(\mathrm{D})}$, which is defined as the composition of the local actions on all nodes $\mathbf{a}_t^{(\mathrm{D})} \triangleq (\mathbf{a}_{1,t}^{(\mathrm{D})}, \ldots, \mathbf{a}_{|\mathcal{V}|,t}^{(\mathrm{D})}) \in \mathcal{A}_\mathrm{D}$, where $\mathcal{A}_\mathrm{D}$ is finite. A local action can be a defensive action or the null action $\perp$ (see examples in Fig. 4). Each defensive action $\mathbf{a}_{i,t}^{(\mathrm{D})} \neq \perp$ leads to $\mathbf{S}_{i,t+1}^{(\mathrm{A})} = (0, 0)$ and may affect $V_{i,t+1}^{(\mathrm{Z})}$.

The defender state $\mathbf{S}_t^{(\mathrm{D})} \triangleq \left(V_{i,t}^{(\mathrm{Z})}\right)_{i \in \mathcal{V}}$ evolves according to

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

where $\mathbf{s}_{t+1}^{(\mathrm{D})}$ is a realization of $\mathbf{S}_{t+1}^{(\mathrm{D})}$.

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.

C. Observability and Strategies

At each time t, the defender and the attacker both observe $\mathbf{o}_t \triangleq \left(\mathbf{o}_{1,t}, \ldots, \mathbf{o}_{|\mathcal{V}|,t}\right) \in \mathcal{O}$, where $\mathcal{O}$ is finite. (In our use

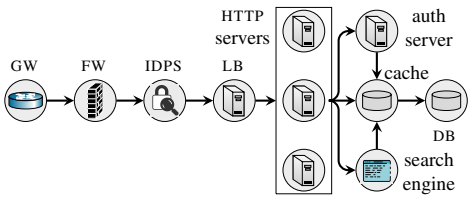


Fig. 5: Dependency graph of a workflow consisting of a chain of virtual network functions and microservices; FW, LB, and IDPS are acronyms for firewall, load balancer, and intrusion detection and prevention system, respectively.

case $\mathbf{o}_t$ relates to the number of IDPS alerts per node.) $\mathbf{o}_t$ 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 $\mathbf{S}_{i,t+1} \triangleq (\mathbf{S}^{(\mathrm{D})}_{i,t+1}, \mathbf{S}^{(\mathrm{A})}_{i,t+1})$. 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 $\mathbf{o} \in \mathcal{O}$.

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

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

where $\gamma \in [0, 1)$ is a discount factor, $c^{(\mathrm{I})}_{i,t}$ is the intrusion cost associated with node i at time t, 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 proportional to the number of active nodes in the subtree rooted at i and that $c^{(\mathrm{I})}_{i,t} = V^{(\mathrm{I})}_{i,t} + c^{(\mathrm{A})}(\mathbf{a}^{(\mathrm{D})}_{i,t})$, where $c^{(\mathrm{A})}$ is a non-negative function.

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

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

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:

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

Solving (6) corresponds to finding a Nash equilibrium [42, 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) [43, 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. $(\mathcal{S}_\mathrm{k})_{\mathrm{k} \in \mathcal{N}}$ are the state spaces, $(\mathcal{A}_\mathrm{k})_{\mathrm{k} \in \mathcal{N}}$ are the action spaces, and $\mathcal{O}$ is observation space (as defined in §IV). 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 $u(\mathbf{s}_t, \mathbf{a}^{(\mathrm{D})}_t)$ 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 histories, 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 sampled from f_D; (*iii*) $\mathbf{s}^{(\mathrm{A})}_{t+1}$ is sampled from f_A; (*iv*) the defender receives the utility $u(\mathbf{s}_t, \mathbf{a}^{(\mathrm{D})}_t)$; and (*v*) the attacker receives the utility $-u(\mathbf{s}_t, \mathbf{a}^{(\mathrm{D})}_t)$.

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 $\mathbf{b}^{(\mathrm{D})}_t(\mathbf{s}^{(\mathrm{A})}_t) \triangleq \mathbb{P}[\mathbf{s}^{(\mathrm{A})}_t \mid \mathbf{H}^{(\mathrm{D})}_t]$ and $\mathbf{b}^{(\mathrm{A})}_t(\mathbf{s}^{(\mathrm{D})}_t) \triangleq \mathbb{P}[\mathbf{s}^{(\mathrm{D})}_t \mid \mathbf{H}^{(\mathrm{A})}_t]$. The belief states are updated at each time $t > 1$ via [43, Eq. 1] and are realizations of $\mathbf{B}^{(\mathrm{D})}_t$ and $\mathbf{B}^{(\mathrm{A})}_t$. The initial beliefs at $t = 1$ are the degenerate distributions $\mathbf{b}^{(\mathrm{D})}_1(\mathbf{0}_{2|\mathcal{V}|}) = 1$ and $\mathbf{b}^{(\mathrm{A})}_1(\mathbf{s}^{(\mathrm{D})}_1) = 1$, where $\mathbf{0}_n$ is the n-dimensional zero-vector and $\mathbf{s}^{(\mathrm{D})}_1$ is given by the infrastructure configuration (see §IV).

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

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

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

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 [42, Eq. 1].

We know from game theory that Γ has a mixed Nash equilibrium [43]–[45] and we know from Markov decision theory that $B_{\mathrm{D}}(\pi_{\mathrm{A}})$ and $B_{\mathrm{A}}(\pi_{\mathrm{D}})$ are non-empty [26], [46]. 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) in Γ with the instantiation described in §IV 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 [44, §3], [45, Thm. 2.3], and [43, Thm. 1] apply, which show that Γ can be modeled as a finite strategic game, for which Nash's theorem applies [42, 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. It then follows from Markov decision theory that a pair of pure best response strategies $(\tilde{\pi}_{\mathrm{D}}, \tilde{\pi}_{\mathrm{A}})$ exists [46, Thm. 6.2.7] [26, 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 [47, 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 [25, Ch. 15], which means that a best response (8)–(9) of the original game can be obtained from best responses of the subgames. We further show that best responses of the subgames can be computed in parallel and that the space complexity of a subgame is independent of the number of nodes $|\mathcal{V}|$. Note that the space complexity of the original game increases exponentially with $|\mathcal{V}|$ (see Fig. 6).

Notation(s)	*Description*
$\mathcal{G}, \mathcal{G}_{\mathbf{w}}$,	Infrastructure tree, subtree of $\mathbf{w}$
$\mathcal{V}, \mathcal{E}$,	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
$\mathcal{A}_{\mathrm{D}}, \mathcal{A}_{\mathrm{A}}(\mathbf{s}_t)$	Defender and attacker action spaces at time t
$\mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}, \mathcal{A}_{\mathrm{A}}^{(\mathrm{V})}(\mathbf{s}_t)$	Action spaces per node at time t, $\mathcal{A}_{\mathrm{k}} = (\mathcal{A}_{\mathrm{k}}^{(\mathrm{V})})^{\lvert\mathcal{V}\rvert}$
$\mathcal{O}^{(\mathrm{V})}$	Observation space per node at time t, $\mathcal{O} = (\mathcal{O}^{(\mathrm{V})})^{\lvert\mathcal{V}\rvert}$
$v_{i,t}^{(\mathrm{I})}, v_{i,t}^{(\mathrm{Z})}, v_{i,t}^{(\mathrm{R})}$	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
$V_{i,t}^{(\mathrm{I})}, V_{i,t}^{(\mathrm{Z})}, V_{i,t}^{(\mathrm{R})}$	Random variables with realizations $v_{i,t}^{(\mathrm{I})}, v_{i,t}^{(\mathrm{Z})}, v_{i,t}^{(\mathrm{R})}$
$\Gamma, \mathcal{N}$	PO-POSG (7), set of players
$u, \mathcal{S}, \mathcal{O}$	Utility function, state space, observation space
$\mathbf{s}_t = (\mathbf{s}_t^{(\mathrm{D})}, \mathbf{s}_t^{(\mathrm{A})})$	State at time t
$\mathbf{a}_t = (\mathbf{a}_t^{(\mathrm{D})}, \mathbf{a}_t^{(\mathrm{A})})$	Action at time t
$\mathbf{o}_t, \mathbf{u}_t, \mathbf{a}_t^{(\mathrm{k})}, \mathbf{h}_t^{(\mathrm{k})}$	Observation and utility at time t
$\mathbf{a}_t^{(\mathrm{k})}, \mathbf{h}_t^{(\mathrm{k})}$	Action and history of player kat time t
$\mathcal{B}_{\mathrm{k}}, \mathbf{b}_t^{(\mathrm{k})}$	Belief space and belief state of player k
$\tilde{\pi}_{\mathrm{k}}, \tilde{\mathbf{a}}^{(\mathrm{k})}$	Best response strategy and action 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 $\mathbf{u}_t, \mathbf{b}_t^{(\mathrm{k})}, \mathbf{h}_t^{(\mathrm{k})}$
π_{k}, Z	Strategy of player k, observation distribution
$u_{i,t}^{(\mathrm{w})}$	Workflow utility of node i at time t
$\perp, \mathrm{an}(i)$,	Null action, set of i and its ancestors in $\mathcal{G}$
$\alpha_{i,t}$	Active status of node i at time t
$f_{\mathrm{A}}, f_{\mathrm{D}}, B_{\mathrm{k}}$	Attacker and defender transition functions
B_{k}	Best response correspondence of player k
$c_{i,t}^{(\mathrm{I})}$	Intrusion cost associated with node i at time t
$c^{(\mathrm{A})}$	Action cost function

TABLE 1: Notations for our mathematical model.

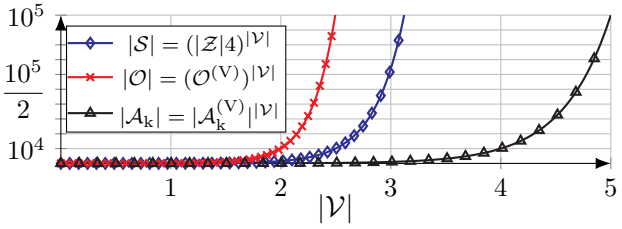


Fig. 6: Growth of $|\mathcal{S}|$, $|\mathcal{O}|$, and $|\mathcal{A}_{\mathrm{k}}|$ in function of the number of nodes $|\mathcal{V}|$, where $\mathrm{k} \in \{\mathrm{D}, \mathrm{A}\}$; the curves are computed using $|\mathcal{Z}| = 10$, $|\mathcal{O}^{(\mathrm{V})}| = 100$, and $|\mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}|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

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 $(\Gamma^{(i)})_{i \in \mathcal{V}_{\mathbf{w}}}$ with optimal substructure and space complexities independent of $|\mathcal{V}|$.

(C) For each subgame $\Gamma^{(i)}$, a best response strategy for the defender can be characterized by switching curves, under the assumption that the observation 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 *[26, Def. 10.2.1]).*

Statements A and B express that Γ decomposes into 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 switching curves, 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. Proof of Theorem 2.A

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

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

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

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 observation-independent, transition-independent, and utility-independent [31, Defs. 32,33,35].

From (3) we have

$$Z\big(\mathbf{O}_{i,t+1} \mid \mathbf{S}_{t+1}^{(\mathrm{D})}, \mathbf{S}_{t+1}^{(\mathrm{A})}\big) = Z\big(\mathbf{O}_{i,t+1} \mid \mathbf{S}_{i,t+1}^{(\mathrm{D})}, \mathbf{S}_{i,t+1}^{(\mathrm{A})}\big) \quad (12)$$

which implies observation independence across nodes $i \in \mathcal{V}$ and therefore across workflows [31, Def. 33].

From the definitions in §IV and (1)–(2) 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 [31, Def. 32].

Following (4) and the definition of $u_{i,t}^{(\mathrm{W})}$ (see §IV-D) we can rewrite $u(\mathbf{s}_t, \mathbf{a}_t^{(\mathrm{D})})$ as

$$\begin{aligned} u(\mathbf{s}_t, \mathbf{a}_t^{(\mathrm{D})}) &= \sum_{\mathbf{w} \in \mathcal{W}} \overbrace{\sum_{i \in \mathcal{V}_{\mathbf{w}}} \eta u_{i,t}^{(\mathrm{W})} - c_{i,t}^{(\mathrm{I})}(\mathbf{a}_{i,t}^{(\mathrm{D})}, V_{i,t}^{(\mathrm{I})})}^{\triangleq u_{\mathbf{w}}} \\ &= \sum_{\mathbf{w} \in \mathcal{W}} u_{\mathbf{w}}\Big((\mathbf{s}_{i,t}, \mathbf{a}_{i,t}^{(\mathrm{D})})_{i \in \mathcal{V}_{\mathbf{w}}}\Big) \end{aligned} \quad (13)$$

The final expression in (13) is a sum of workflow 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 [31, Def. 35]. □

B. Proof of Theorem 2.B

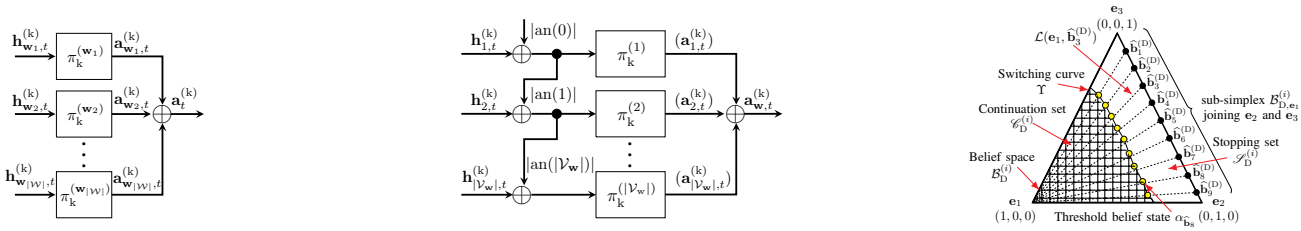
Our goal is to show that a workflow subgame $\Gamma^{(\mathbf{w})}$ decomposes into node-level subgames with optimal substructure. That is, we aim to show that a best response in $\Gamma^{(\mathbf{w})}$ can be constructed from best responses of the subgames.

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 $(\Gamma^{(i)})_{i \in \mathcal{V}_{\mathbf{w}}}$ where each subgame depends only on the local state and action of a single node. 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 and observation-independent but utility-dependent. To prove optimal substructure it therefore suffices to show that it is possible to redefine the utility functions for the subgames such that at each time t, the best response action in $\Gamma^{(\mathbf{w})}$ for any node i is also a best response in $\Gamma^{(i)}$ and vice versa. For the sake of brevity we give the proof for the defender only. The proof for the attacker is analogous. In this proof, for better readability, we omit the constants γ, η and use the shorthand notations $\mathbf{s}_{\mathbf{w},t}^{(\mathrm{D})} \triangleq (\mathbf{s}_{j,t}^{(\mathrm{D})})_{j \in \mathcal{V}_{\mathbf{w}}}$, $\mathbf{b}_{\mathbf{w},t}^{(\mathrm{D})} \triangleq (\mathbf{b}_{j,t}^{(\mathrm{D})})_{j \in \mathcal{V}_{\mathbf{w}}}$, $\mathscr{V} \triangleq \mathscr{V}_{\mathrm{D},\pi_{\mathrm{A}}}^{*}$, and $\tau \in \arg\min_{k > t} \mathbf{a}_k^{(\mathrm{D})} \neq \bot$, where $\mathscr{V}$ is the value function [26, Thm. 7.4.1]. Further, we use $\mathrm{an}(i)$ to denote the set of node i and its ancestors in the infrastructure graph $\mathcal{G}$.

From Bellman's optimality equation [48, Eq. 1] a best response action for node i at time t in $\Gamma^{(\mathbf{w})}$ against an attacker strategy π_{A} is given by

$$\begin{aligned}
&\underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[\mathbf{U}_t + \mathscr{V}(\mathbf{S}_{t+1}^{(\mathrm{D})}, \mathbf{B}_{t+1}^{(\mathrm{D})}) \middle| \mathbf{s}_t^{(\mathrm{D})}, \mathbf{b}_t^{(\mathrm{D})}, \mathbf{a}_{i,t}^{(\mathrm{D})} \right] \right] \\
&\overset{(a)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[-c_{i,t}^{(\mathrm{I})} + \mathscr{V}(\mathbf{S}_{t+1}^{(\mathrm{D})}, \mathbf{B}_{t+1}^{(\mathrm{D})}) \middle| \mathbf{s}_t^{(\mathrm{D})}, \mathbf{b}_t^{(\mathrm{D})}, \mathbf{a}_{i,t}^{(\mathrm{D})} \right] \right] \\
&\overset{(b)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[-c_{i,t}^{(\mathrm{I})} + \sum_{k=t+1}^{\infty} \sum_{j \in \mathcal{V}_{\mathbf{w}}} \mathbf{U}_{j,k} \middle| \overbrace{\mathbf{s}_{\mathbf{w},t}^{(\mathrm{D})}, \mathbf{b}_{\mathbf{w},t}^{(\mathrm{D})}, \mathbf{a}_{i,t}^{(\mathrm{D})}}^{\triangleq \kappa} \right] \right] \\
&\overset{(c)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[-c_{i,t}^{(\mathrm{I})} + \sum_{k=t+1}^{\tau} \sum_{j \in \mathcal{V}_{\mathbf{w}}} \mathbf{U}_{j,k} \middle| \kappa \right] \right] \\
&\overset{(d)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[-c_{i,t}^{(\mathrm{I})} + \sum_{k=t+1}^{\tau} \sum_{j \in \mathrm{an}(i)} \mathbf{U}_{j,k} \middle| \kappa \right] \right] \\
&\overset{(e)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[-c_{i,t}^{(\mathrm{I})} + \sum_{k=t+1}^{\tau} \sum_{j \in \mathrm{an}(i)} u_{j,k}^{(\mathrm{W})} - c_{j,k}^{(\mathrm{I})} \middle| \kappa \right] \right] \\
&\overset{(f)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[\overbrace{-c_{i,t}^{(\mathrm{I})} + \sum_{k=t+1}^{\tau} |\mathrm{an}(i)| \alpha_{i,t+1} - c_{i,k}^{(\mathrm{I})}}^{\triangleq \omega} \middle| \kappa \right] \right] \\
&\overset{(g)}{=} \underset{\mathbf{a}_{i,t}^{(\mathrm{D})} \in \mathcal{A}_{\mathrm{D}}^{(\mathrm{V})}}{\arg\max} \left[\mathbb{E}_{\pi_{\mathrm{A}}} \left[\omega \;\middle|\; \mathbf{s}_{i,t}^{(\mathrm{D})}, \mathbf{b}_{i,t}^{(\mathrm{D})}, \mathbf{a}_{i,t}^{(\mathrm{D})} \right] \right]
\end{aligned} \quad (14)$$

where $\mathbf{U}_t$ denotes the vector of utilities for all nodes at time t. (a) holds because $(\mathbf{U}_{j,t})_{j \in \mathcal{V} \setminus \{i\}}$ and $u_{i,t}^{(\mathrm{W})}$ are independent of $\mathbf{a}_{i,t}^{(\mathrm{D})}$ and therefore does not affect the maximization; (b) follows from the utility independence across workflows (Thm. 2.A) and the definition of the value function $\mathscr{V}$ [26,



(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; $k \in \{\mathrm{D}, \mathrm{A}\}$; $\mathbf{h}^{(\mathrm{k})}_{\mathbf{w},t} \triangleq (\mathbf{h}^{(\mathrm{k})}_{j,t})_{j \in \mathcal{V}_{\mathbf{w}}}$; and $\mathbf{a}^{(\mathrm{k})}_{\mathbf{w},t} \triangleq (\mathbf{a}^{(\mathrm{k})}_{j,t})_{j \in \mathcal{V}_{\mathbf{w}}}$; (a) illustrates that a game strategy π_{k} decomposes into $|\mathcal{W}|$ independent substrategies; (b) illustrates that a workflow strategy $\pi^{(\mathbf{w})}_{\mathrm{k}}$ for $\mathbf{w} \in \mathcal{W}$ decomposes into substrategies $(\pi^{(i)}_{\mathrm{k}})_{i \in \mathcal{V}_{\mathbf{w}}}$ 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 $i \in \mathcal{V}$.

Thm. 7.4.1]; (c) holds because any $\mathbf{a}^{(\mathrm{D})}_{i,t}$ except $\perp$ leads to $\mathbf{s}^{(\mathrm{A})}_{i,t+1} = (0,0)$, which means that all state variables at time $k > \tau$ are independent of $\mathbf{a}^{(\mathrm{D})}_{i,t}$ and can therefore be moved outside the $\arg\max$ operator; (d) follows because $(\mathbf{U}_{j,t})_{j \in \mathcal{V} \setminus \mathrm{an}(i)}$ is independent of $\mathbf{a}^{(\mathrm{D})}_{i,t}$; (e) is an expansion of $(\mathbf{U}_{j,k})_{j \in \mathrm{an}(i), k \in \{t+1,\dots,\tau\}}$ based on (4); and (f)-(g) follow because the terms in $(u^{(\mathrm{W})}_{j,k})_{j \in \mathrm{an}(i), k \in \{t+1,\dots,\tau\}}$ that depend on $\mathbf{a}^{(\mathrm{D})}_{i,t}$ equal $k|\mathrm{an}(i)|\alpha_{t+1,i}$, where k is the constant of proportionality (see §IV). (Recall that $\alpha_{i,t} = 1$ if node i is active at time t and $\alpha_{i,t} = 0$ otherwise.)

The final expression in (14) depends only on local information related to node i. This means that we can use it to define utility functions of the subgames $(\Gamma^{(i)})_{i \in \mathcal{V}_{\mathbf{w}}}$ such that they become utility-independent. Further, since the maximizer of the final expression in (14) is also a maximizer of the first expression, it follows that a a best response in $\Gamma^{(i)}$ is also a best response for node i in $\Gamma^{(\mathbf{w})}$ and thus in Γ (Thm. 2.A). Hence $(\Gamma^{(i)})_{i \in \mathcal{V}_{\mathbf{w}}}$ have optimal substructure. □

C. Proof of Theorem 2.C

The idea behind this proof is that the problem of selecting *which* defensive action to apply in a subgame $\Gamma^{(i)}$ (Thm. 2.B) against a given attacker strategy 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 [26, Thm. 12.3.4], the optimal stopping strategy in $\Gamma^{(i)}$ can be characterized by switching curves.

We perform the above separation by decomposing $\mathbf{a}^{(\mathrm{D})}_{i,t}$ into two subactions: $\mathbf{a}^{(\mathrm{D},1)}_{i,t}$ and $\mathbf{a}^{(\mathrm{D},2)}_{i,t}$ which realize $\mathbf{A}^{(\mathrm{D},1)}_{i,t}$ and $\mathbf{A}^{(\mathrm{D},2)}_{i,t}$. The first subaction $\mathbf{a}^{(\mathrm{D},1)}_{i,t} \neq \perp$ determines the defensive action and the second subaction $\mathbf{a}^{(\mathrm{D},2)}_{i,t} \in \{\mathrm{S}, \mathrm{C}\}$ determines when to take it. Specifically, if $\mathbf{a}^{(\mathrm{D},2)}_{i,t} = \mathrm{C}$, then $\mathbf{a}^{(\mathrm{D})}_{i,t} = \perp$, otherwise $\mathbf{a}^{(\mathrm{D})}_{i,t} = \mathbf{a}^{(\mathrm{D},1)}_{i,t}$. Using this action decomposition, at each time t, a strategy $\pi^{(i)}_{\mathrm{D}}$ in $\Gamma^{(i)}$ is a joint distribution over $\mathbf{A}^{(\mathrm{D},1)}_{i,t}$ and $\mathbf{A}^{(\mathrm{D},2)}_{i,t}$, which means that it can be represented in an auto-regressive manner as

$$\pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},1)}_{i,t}, \mathbf{A}^{(\mathrm{D},2)}_{i,t} \mid \mathbf{H}^{(\mathrm{k})}_{i,t}) \tag{15}$$

$$\overset{(a)}{=} \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},1)}_{i,t} \mid \mathbf{H}^{(\mathrm{D})}_{i,t}) \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},2)}_{i,t} \mid \mathbf{H}^{(\mathrm{D})}_{i,t}, \mathbf{A}^{(\mathrm{D},1)}_{i,t})$$
$$\overset{(b)}{=} \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},1)}_{i,t} \mid \mathbf{B}^{(\mathrm{D})}_{i,t}, \mathbf{S}^{(\mathrm{D})}_{i,t}) \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},2)}_{i,t} \mid \mathbf{B}^{(\mathrm{D})}_{i,t}, \mathbf{S}^{(\mathrm{D})}_{i,t}, \mathbf{A}^{(\mathrm{D},1)}_{i,t})$$
$$\overset{(c)}{=} \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},1)}_{i,t} \mid \mathbf{S}^{(\mathrm{D})}_{i,t}) \pi^{(i)}_{\mathrm{D}}(\mathbf{A}^{(\mathrm{D},2)}_{i,t} \mid \mathbf{B}^{(\mathrm{D})}_{i,t}, \mathbf{S}^{(\mathrm{D})}_{i,t}, \mathbf{A}^{(\mathrm{D},1)}_{i,t})$$

where (a) follows from the chain rule of probability; (b) holds because $(\mathbf{S}^{(\mathrm{D})}_{i,t}, \mathbf{B}^{(\mathrm{D})}_{i,t})$ is a sufficient statistic for $\mathbf{H}^{(\mathrm{D})}_{i,t}$ [26, Thm 7.2.1]; and (c) follows because

$$\begin{aligned} \operatorname*{arg\,max}_{\mathbf{a}^{(\mathrm{D},1)}_{i,t} \in \mathcal{A}^{(\mathrm{V})}_{\mathrm{D}} \setminus \perp} \Bigg[& \eta |\mathrm{an}(i)| \alpha_{i,t+1} - c^{(\mathrm{I})}_{i,t}(V^{(\mathrm{I})}_{i,t}, \mathbf{a}^{(\mathrm{D},1)}_{i,t}) + \\ & \gamma \mathbb{E}\Big[\mathscr{V}(\mathbf{S}^{(\mathrm{D})}_{i,t+1}, \mathbf{B}^{(\mathrm{D})}_{i,t+1}) \;\Big|\; \mathbf{s}^{(\mathrm{D})}_{i,t}, \mathbf{b}^{(\mathrm{D})}_{i,t}, \mathbf{a}^{(\mathrm{D},1)}_{i,t} \Big] \Bigg] \end{aligned} \tag{16}$$

$$\begin{aligned} \overset{(a)}{=} \operatorname*{arg\,max}_{\mathbf{a}^{(\mathrm{D},1)}_{i,t} \in \mathcal{A}^{(\mathrm{V})}_{\mathrm{D}} \setminus \perp} \Bigg[& \eta |\mathrm{an}(i)| \alpha_{i,t+1} - c^{(\mathrm{A})}(\mathbf{a}^{(\mathrm{D},1)}_{i,t}) + \\ & \gamma \mathbb{E}\Big[\mathscr{V}(\mathbf{S}^{(\mathrm{D})}_{i,t+1}, \mathbf{B}^{(\mathrm{D})}_{i,t+1}) \;\Big|\; \mathbf{s}^{(\mathrm{D})}_{i,t}, \mathbf{b}^{(\mathrm{D})}_{i,t}, \mathbf{a}^{(\mathrm{D},1)}_{i,t} \Big] \Bigg] \end{aligned}$$

$$\begin{aligned} \overset{(b)}{=} \operatorname*{arg\,max}_{\mathbf{a}^{(\mathrm{D},1)}_{i,t} \in \mathcal{A}^{(\mathrm{V})}_{\mathrm{D}} \setminus \perp} \Bigg[& \eta |\mathrm{an}(i)| \alpha_{i,t+1} - c^{(\mathrm{A})}(\mathbf{a}^{(\mathrm{D},1)}_{i,t}) \\ & \gamma \mathbb{E}\Big[\mathscr{V}(\mathbf{S}^{(\mathrm{D})}_{i,t+1}, \mathbf{e}_1) \;\Big|\; \mathbf{s}^{(\mathrm{D})}_{i,t}, \mathbf{a}^{(\mathrm{D},1)}_{i,t} \Big] \Bigg] \end{aligned}$$

which means that $\mathbf{a}^{(\mathrm{D},1)}_{i,t} \neq \perp$ is independent of $\mathbf{B}^{(\mathrm{D})}_{i,t}$. The first statement in (16) is the Bellman equation [48, Eq. 1]; (a) holds because $V^{(\mathrm{I})}_{i,t}$ is independent of $\mathbf{a}^{(\mathrm{D},1)}_{i,t}$; and (b) is true because any $\mathbf{a}^{(\mathrm{D},1)}_{i,t} \neq \perp$ leads to $\mathbf{S}^{(\mathrm{A})}_{i,t+1} = (0,0)$ and thus to $\mathbf{B}^{(\mathrm{D})}_{i,t+1} = \mathbf{e}_1 = (1,0,0)$. (Recall that the belief space $\mathcal{B}^{(i)}_{\mathrm{D}}$ is the two-dimensional unit simplex.)

The strategy decomposition in (15) means that we can obtain a best response strategy in $\Gamma^{(i)}$ by jointly optimizing two substrategies: $\pi^{(i,1)}_{\mathrm{D}}$ and $\pi^{(i,2)}_{\mathrm{D}}$. The former corresponds to solving an MDP $\mathscr{M}^{(\mathrm{D},1)}$ with state space $\mathbf{s}^{(\mathrm{D})}_i \in \mathcal{Z}$ and the latter corresponds to solving a set of optimal stopping POMDPs $(\mathscr{M}^{(\mathrm{D},2)}_{i,s^{(\mathrm{D})},a^{(\mathrm{D})}})_{s^{(\mathrm{D})} \in \mathcal{Z}, a^{(\mathrm{D})} \in \mathcal{A}^{(\mathrm{V})}_{\mathrm{D}}}$ with state space $\mathbf{s}^{(\mathrm{A})}_i \in \{(0,0),(1,0),(1,1)\}$.

Each stopping problem can be defined with a *single* stop action rather than multiple stop actions [3, §III.C] because

$$\begin{aligned}
&\arg\max_{\pi_{\mathrm{D}}\in\Pi_{\mathrm{D}}^{(i,2)}}\left[\mathbb{E}_{\pi_{\mathrm{D}}}\left[\sum_{t=1}^{\infty}\gamma^{t-1}\mathbf{U}_{i,2,t}\;\middle|\;\mathbf{B}_{i,1}^{(\mathrm{D})}=\mathbf{e}_1\right]\right]\\
&=\arg\max_{\pi_{\mathrm{D}}\in\Pi_{\mathrm{D}}^{(i,2)}}\left[\mathbb{E}_{\pi_{\mathrm{D}}}\left[\sum_{t=1}^{\tau_1}\gamma^{t-1}\mathbf{U}_{i,2,t}\;\middle|\;\mathbf{B}_{i,1}^{(\mathrm{D})}=\mathbf{e}_1\right]+\right.\\
&\qquad\left.\mathbb{E}_{\pi_{\mathrm{D}}}\left[\sum_{t=\tau_1+1}^{\tau_2}\gamma^{t-1}\mathbf{U}_{i,2,t}\;\middle|\;\mathbf{B}_{i,\tau_1+1}^{(\mathrm{D})}=\mathbf{e}_1\right]+\ldots\right]\\
&=\arg\max_{\pi_{\mathrm{D}}\in\Pi_{\mathrm{D}}^{(i,2)}}\left[\mathbb{E}_{\pi_{\mathrm{D}}}\left[\sum_{t=1}^{\tau_1}\gamma^{t-1}\mathbf{U}_{i,2,t}\;\middle|\;\mathbf{B}_{i,1}^{(\mathrm{D})}=\mathbf{e}_1\right]\right] \qquad (17)
\end{aligned}$$

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 $\mathscr{M}_{i,s^{(\mathrm{D})},a^{(\mathrm{D})}}^{(\mathrm{D},2)}$. Note that the belief space $\mathcal{B}_{\mathrm{D}}^{(i)}$ for each stopping problem is the 2-dimensional unit simplex and that $\mathbf{B}_{i,\tau_j+1}^{(\mathrm{D})}=\mathbf{e}_1=(1,0,0)$ for each stopping time τ_j since $\mathbf{a}_{i,\tau_j}^{(\mathrm{D},2)}=\mathrm{S}\implies\mathbf{s}_{i,\tau_j+1}^{(\mathrm{A})}=(0,0)$.

The transition matrices for each stopping problem are of the form:

$$\begin{bmatrix}1-p & p & 0\\ 0 & 1-q & q\\ 0 & 0 & 1\end{bmatrix}\quad\text{and}\quad\begin{bmatrix}1 & 0 & 0\\ 1 & 0 & 0\\ 1 & 0 & 0\end{bmatrix} \qquad (18)$$

where p is the probability that the attacker performs reconnaissance and q is the probability that the attacker compromises the node. The left matrix in (18) relates to $\mathbf{a}_{i,t}^{(\mathrm{D},2)}=\mathrm{C}$ and the right matrix relates to $\mathbf{a}_{i,t}^{(\mathrm{D},2)}=\mathrm{S}$. The non-zero second order minors of the matrices are $(1-p)(1-q)$, pq, $1-q$, $1-p$, p, and $(1-p)q$, which implies that the matrices are TP-2 [26, 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 [26, Thm. 12.3.4] that there exists a switching curve Υ that partitions $\mathcal{B}_{\mathrm{D}}^{(i)}$ into two individually connected regions: a stopping set $\mathscr{S}_{\mathrm{D}}^{(i)}$ where $\mathbf{a}_{i,t}^{(\mathrm{D},2)}=\mathrm{S}$ is a best response and a continuation set $\mathscr{C}_{\mathrm{D}}^{(i)}$ where $\mathbf{a}_{i,t}^{(\mathrm{D},2)}=\mathrm{C}$ is a best response (see Fig. 7c).

The argument behind the existence of a switching curve is as follows [26, Thm. 12.3.4]. 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 with respect to the Monotone Likelihood Ratio (MLR) order [26, Def. 10.1.1]. As a consequence, Topkis's theorem [49, 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. Consequently, 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 $\alpha_{\widehat{\mathbf{b}}_1^{(\mathrm{D})}},\alpha_{\widehat{\mathbf{b}}_2^{(\mathrm{D})}},\ldots$ yield a switching curve Υ. □

VII. Finding Nash Equilibria of the Decomposed Intrusion Response Game

To find a Nash equilibrium of Γ (7) we develop a *fictitious self-play* algorithm called Decompositional Fictitious Self-Play (DFSP), which estimates Nash equilibria based on the decomposition presented above. The pseudocode is listed in Alg. 1. (In Alg. 1, $\oplus$ denotes vector concatenation, $-\mathrm{k}$ denotes the opponent of player k, and $\mathscr{M}_i^{(\mathrm{k})}$ denotes the best response POMDP of k in $\Gamma^{(i)}$ (Thm 2).)

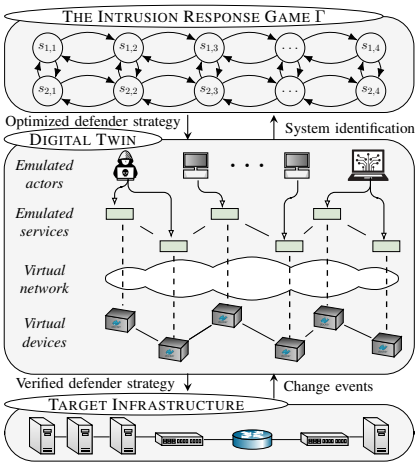


Fig. 8: The digital twin is a virtual replica of the target infrastructure and is used for evaluation and data collection.

Algorithm 1: DFSP

Input: P-SOLVER: a POMDP solver,
δ: convergence criterion, Γ: the PO-POSG
Output: 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, Γ, 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, Γ, k, $\pi_{-\mathrm{k}}$)
  $\boldsymbol{\pi}_{\mathrm{k}}\leftarrow()$
  in parallel for $\mathbf{w}\in\mathcal{W}$, $(i)\in\mathcal{V}_{\mathbf{w}}$ **do**
    $\boldsymbol{\pi}_{\mathrm{k}}\leftarrow\boldsymbol{\pi}_{\mathrm{k}}\oplus$ P-SOLVER($\mathscr{M}_i^{(\mathrm{k})},\pi_{-\mathrm{k}}$)
  return $\boldsymbol{\pi}_{\mathrm{k}}$
Procedure COMPOSITE-STRATEGY($\Gamma,\boldsymbol{\pi}_{\mathrm{k}}$)
  return $\pi_{\mathrm{k}}\leftarrow$**Procedure** λ ($\mathbf{s}_t^{(\mathrm{k})}$, $\mathbf{b}_t^{(\mathrm{k})}$)
    $\mathbf{a}_t^{(\mathrm{k})}\leftarrow()$
    for $\mathbf{w}\in\mathcal{W}$, $i\in\mathcal{V}_{\mathbf{w}}$ **do**
      $\mathbf{a}_t^{(\mathrm{k})}\leftarrow\mathbf{a}_t^{(\mathrm{k})}\oplus(\boldsymbol{\pi}_{\mathrm{k}}^{(i)}(\mathbf{s}_{i,t}^{(\mathrm{k})},\mathbf{b}_{i,t}^{(\mathrm{k})}))$
    end
    return $\mathbf{a}_t^{(\mathrm{k})}$

DFSP implements the fictitious play process described in [50] 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 $(\pi^*_{\mathrm{D}},\pi^*_{\mathrm{A}})$ [51, Thms. 7.2.4–7.2.5]. During each step of this process, DFSP

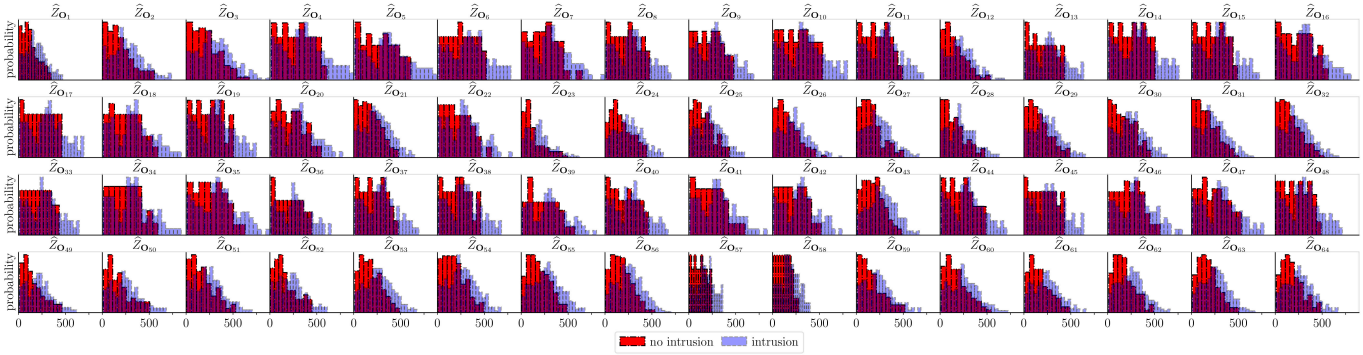


Fig. 9: Empirical observation distributions $\widehat{Z}_{\mathbf{O}_1}, \ldots, \widehat{Z}_{\mathbf{O}_{|\mathcal{V}|}}$ as estimates of $Z_{\mathbf{O}_1}, \ldots, Z_{\mathbf{O}_{|\mathcal{V}|}}$ in the target infrastructure (depicted in Fig. 1); $\mathbf{O}_i$ is a random variable representing the 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).

learns best responses against the players' current strategies and then updates both players' strategies (lines 7–11 in Alg. 1). To obtain the best responses, it first finds best responses for the node subgames as constructed in the proof of Thm. 2.B (lines 14–18), and then it combines them using the method described in §VI-B (lines 19–25).

Finding best responses for node subgames amounts to solving POMDPs. The principal method for solving POMDPs is dynamic programming [26]. Dynamic programming is however intractable in our case, as demonstrated in Fig. 10b. To find the best responses we instead resort to approximation algorithms. More specifically, we use the Proximal Policy Optimization (PPO) algorithm [52, Alg. 1] 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 [26, Eq. 6.21], which can be done efficiently due to full observability. After solving the MDP, we approximate the optimal switching curves defined in the proof of Thm. 2.C (§VI-C) with the following linear approximation [26, Eq. 12.18].

$$\pi_{\mathrm{D}}(\mathbf{b}^{(\mathrm{D})}) = \begin{cases} \mathrm{S} & \text{if } \begin{bmatrix} 0 & 1 & \boldsymbol{\theta} \end{bmatrix} \begin{bmatrix} (\mathbf{b}^{(\mathrm{D})})^T \\ -1 \end{bmatrix} > 0 \\ \mathrm{C} & \text{otherwise} \end{cases} \tag{19}$$

$$\text{subject to } \boldsymbol{\theta} \in \mathbb{R}^2, \ \boldsymbol{\theta}_2 > 0, \ \text{and } \boldsymbol{\theta}_1 \geq 1$$

The coefficients $\boldsymbol{\theta}$ in (19) are estimated through the stochastic approximation algorithm in [26, Alg. 14] and [3, 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. 8). This section describes the digital twin (§VIII-A) and the identification process (§VIII-B).

A. Creating a Digital Twin of the Target Infrastructure

We create a digital twin of the target infrastructure shown in Fig. 1 through an emulation system. Documentation of this emulation system is available in [6].

The process of creating the digital twin involves two main tasks. The first task is to replicate relevant parts of the physical infrastructure that is emulated, such as physical resources, network interfaces, and network conditions. This task is described in §VIII-A1. The second task is to emulate actors in the digital twin (e.g. the attacker, the defender, and the client population). We describe this task in §VIII-A2.

1) Emulating physical resources: The physical resources of the target infrastructure are emulated through the following steps.

Emulating physical hosts. Physical hosts are emulated with Docker containers [53], i.e. lightweight executable packages that include runtime systems, code, system tools, system libraries, and configurations. Resource allocation to containers, e.g. CPU and memory, is enforced using CGROUPs. The software functions running inside the containers replicate important components of the target infrastructure, such as, web servers, databases, the SNORT IDPS [54], and the RYU SDN controller [55].

Emulating physical switches. Physical switches are emulated with Docker containers that run Open vSwitch (OVS) [56] and may connect to a controller through the OPENFLOW protocol [57]. (Since the switches are programmed through flow tables, they can act either as classical layer 2 switches or as routers, depending on the flow table configurations.)

Emulating physical network links. Network connectivity is emulated with virtual links implemented by Linux bridges. Network isolation between virtual containers on the same physical host is achieved through network namespaces, which create logical copies of the physical host's network stack. To connect containers on different physical hosts, the emulated traffic is tunneled over the physical network using VXLAN tunnels [58].

Emulating network conditions. Network conditions of virtual links are configured using the NETEM module in the Linux kernel [59]. This module allows fine-gained configuration of

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 2: Attacker actions in the digital twin; exploits are identified according to the Common Vulnerabilities and Exposures (CVE) database [63] and the Common Weakness Enumeration (CWE) list [64].

bit rates, packet delays, packet loss probabilities, jitter, and packet reordering probabilities.

We emulate connections between servers as full-duplex lossless connections of 1 Gbit/s capacity in both directions. We emulate connections between the gateway and the external client population as full-duplex connections of 100 Mbit/s capacity and 0.1% packet loss with random bursts of 1% packet loss. (These numbers are based on measurements on enterprise and wide-area networks [60]–[62].)

2) Emulating Actors in the Digital Twin: In this section, we describe how actors of the intrusion response use case described in §III are emulated in the digital twin.

Emulating the client population. The *client population* is emulated by processes in Docker containers. Clients interact with application servers through the gateway by consuming workflows. The workflow of a client is selected uniformly at random and its sequence of service invocations is decided uniformly at random. Client arrivals per time-step are emulated using a stationary Poisson process with rate $\lambda = 50$ and exponentially distributed service times with mean $\mu = 4$. The duration of a time-step is 30 seconds.

Emulating the attacker. The attacker's actions are emulated by executing scripts that automate exploits (see Table 2).

Emulating the defender. The four types of defender actions (see Fig. 4) are emulated as follows. To emulate the *server migration* action, we remove all virtual network interfaces of the emulated server and add a new interface that connects it to the new zone. To emulate the *flow migration/blocking* action we add rules to the flow tables of the emulated switches that match all flows towards the server and redirect them to a given destination. To emulate the *server shut down* action, we shut down the virtual container corresponding to the emulated server. Finally, to emulate the *access control* action, we reset all user accounts and certificates on the emulated server.

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. 1), 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 [54]. (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. 9).

We observe in Fig. 9 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. 2). 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 on a digital twin of the target infrastructure and in simulations of synthetic infrastructures. The topology of the target infrastructure is depicted in Fig. 1 and its configuration is available in App. C. The digital twin 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 CPUS. Code for replicating the experiments is available in [6].

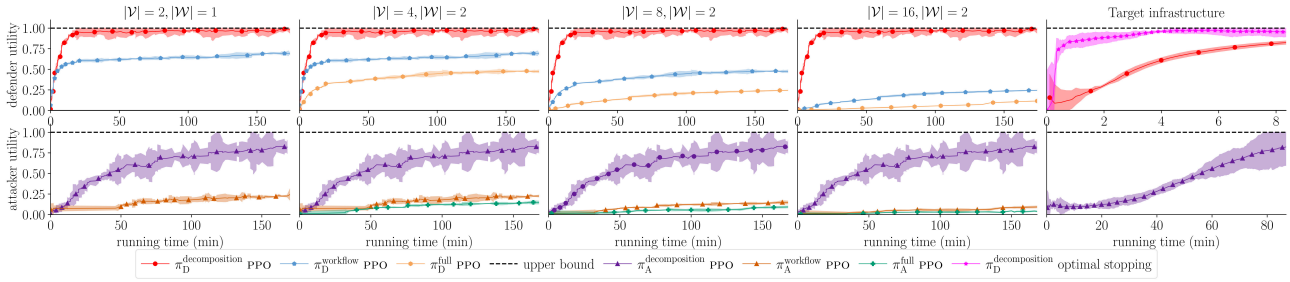
Convergence metric. To estimate the convergence of the sequence of strategy pairs generated by DFSP, we use the *approximate exploitability* metric $\widehat{\delta}$ [66]:

$$\widehat{\delta} = \mathbb{E}_{\widehat{\pi}_{\mathrm{D}}, \pi_{\mathrm{A}}}[J] - \mathbb{E}_{\pi_{\mathrm{D}}, \widehat{\pi}_{\mathrm{A}}}[J] \tag{20}$$

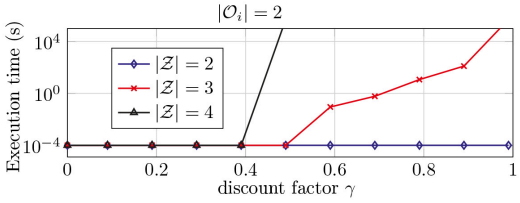
where J is defined in (4) and $\widehat{\pi}_{\mathrm{k}}$ denotes an approximate best response strategy for player k. The closer $\widehat{\delta}$ becomes to 0, the closer $(\pi_{\mathrm{D}}, \pi_{\mathrm{A}})$ is to a Nash equilibrium.

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.

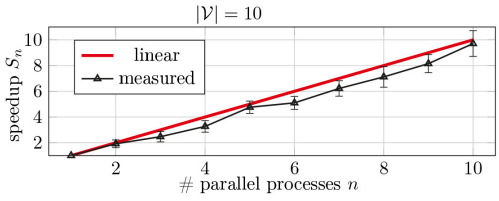
We compare the performance of DFSP with that of Neural Fictitious Self-Play (NFSP) [67, Alg. 1] and PPO [52, Alg.1], which are the most popular algorithms among related work (see [19, §VII] for a review of algorithms used in related work).



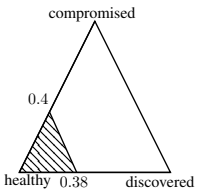
(a) Best response learning curves for the target infrastructure and synthetic infrastructures with varying $|\mathcal{V}|$ and $|\mathcal{W}|$.



(b) Runtimes of dynamic programming.



(c) Best response scalability.



(d) Best response structure.

Fig. 10: 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 [65]; (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).

Baseline strategies. We compare the defender strategies learned through DFSP with three baselines. The first baseline selects actions uniformly at random. The second baseline assumes prior knowledge of the opponent's actions and acts optimally based on this information. The last baseline acts according to the following heuristic: shut down a node $i \in \mathcal{V}$ when 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 $|\mathcal{W}|$. To evaluate the optimal stopping approach described in §VII we compare its rate of convergence with that of PPO. Figure 10a shows the learning curves. The red, purple, and pink curves represent the results obtained with $\pi^{\text{decomposition}}$; the blue and 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.

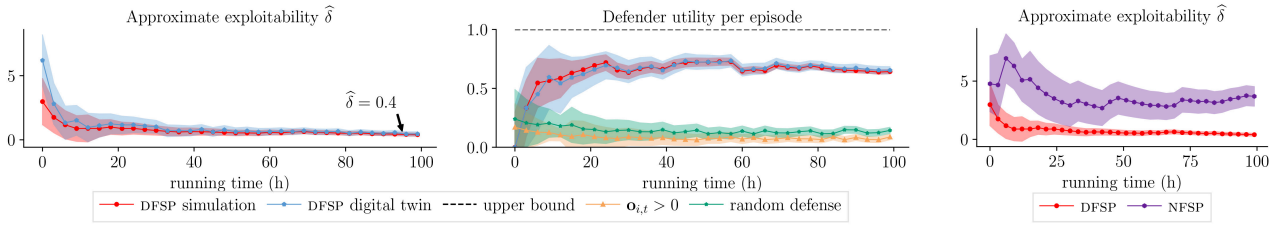
We note that all the learning curves of $\pi^{\text{decomposition}}$ converge near the dashed black lines, which suggests that the learned strategies are close to best responses. In contrast, the learning curves of π^{workflow} and π^{full} do not converge near the dashed black lines within the measured 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. 10c.) Lastly, we note in the rightmost plot of Fig. 10a 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 based on the linear approximation in (19) is shown in Fig. 10d.

B. Learning Equilibrium Strategies through Self-Play

Figures 11a–11b show 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. 11a converge, which suggests that the learned strategies converge as well. Specifically, we observe that the approximate exploitability (20) 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.

Figure 11b compares DFSP with NFSP on the simulator. NFSP implements fictitious self-play and can thus be compared with DFSP with respect to approximate exploitability (20). We observe that DFSP converges significantly faster than NFSP. The fast convergence of DFSP in comparison with NFSP is expected as DFSP is parallelizable while NFSP is not.



(a) Nash equilibrium learning curves.

(b) Comparison DFSP and NFSP.

Fig. 11: 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 (20) and normalized utility; the curves indicate the mean and the shaded areas indicate the standard deviation over three random seeds.

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 approximates optimal defender strategies for a practical IT infrastructure (see Fig. 11a). While we have not evaluated the learned strategies on the target infrastructure due to safety reasons, the fact that they achieve almost the same performance on the digital twin as on the simulator gives us confidence in the strategies performance on the target infrastructure.

(ii) Decomposition provides a scalable approach to automate intrusion responses for IT infrastructures (see Fig. 10a and Fig. 11b). 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. 10a). 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 learned through DFSP improve over time (see the right plot in Fig. 11a). This finding is consistent with previous studies that use game-theoretic approaches (e.g. [36] and [19]) and suggests limitations of static intrusion response systems, such as the Snort IDPS.

X. Conclusions and Future Work

We combine game theory, game decomposition, reinforcement learning, and a digital twin in a framework to address the problem of automated intrusion response for a realistic use case. We formalize the use case as a partially observed stochastic game. We prove a decomposition theorem stating that the game decomposes recursively into subgames that can be solved in parallel and that the best response defender strategies exhibit threshold structures. This decomposition provides us with a 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 for a target infrastructure, we evaluate them on a digital twin. The results demonstrate that DFSP converges in reasonable time to near-optimal strategies both in simulation and on the digital twin while a state-of-the-art algorithm makes little progress toward an optimal strategy within the same time frame.

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. 1 by disregarding the redundant edges in the R&D zone. The remaining parameters are listed in Table 3.

Appendix B
Static Defender and Attacker Strategies

The static defender and attacker strategies for the evaluation described in §IX-A are defined in (21)–(22). (w.p is short for "with probability".)

$$\pi_{\mathrm{D}}(\mathbf{h}_t^{(\mathrm{D})})_i = \begin{cases} \perp & \text{w.p } 0.95 \\ j \in \mathcal{Z} & \text{w.p } \dfrac{0.05}{|\mathcal{Z}| + 1} \end{cases} \tag{21}$$

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. 1
$\lvert\mathcal{V}_{\mathbf{w}_1}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_2}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_3}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_4}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_5}\rvert$,$\lvert\mathcal{V}_{\mathbf{w}_6}\rvert$	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}, $4\cdot 10^3 t$, 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}, $5\cdot 10^{-3}$, 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	
$c,\epsilon,\lambda,A,a,N,\delta$	10, 0.101, 0.602, 100, 1, 50, 0.2

TABLE 3: Hyperparameters ($[\cdot]$ is the Iverson bracket).

$$\pi_{\mathrm{A}}(\mathbf{h}_t^{(\mathrm{A})})_i = \begin{cases} \perp & \text{if } v_{i,t}^{(\mathrm{I})}=1 \\ \perp & \text{w.p } 0.8 \text{ if } v_{i,t}^{(\mathrm{R})}=0 \\ \perp & \text{w.p } 0.7 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0 \\ \text{recon} & \text{w.p } 0.2 \text{ if } v_{i,t}^{(\mathrm{R})}=0 \\ \text{brute} & \text{w.p } 0.15 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0 \\ \text{exploit} & \text{w.p } 0.15 \text{ if } v_{i,t}^{(\mathrm{R})}=1, v_{i,t}^{(\mathrm{I})}=0 \end{cases} \tag{22}$$

Appendix C
Configuration of the Infrastructure in Fig. 1

The configuration of the target infrastructure (Fig. 1) is available in Tables 4 and 5.

References

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

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

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

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

[5] ——, "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.

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

[7] 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.

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

[9] 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.

[10] 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.

[11] 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.

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

[13] 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.

[14] 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.

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

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

[17] 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.

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

[19] K. Hammar and R. Stadler, "Learning near-optimal intrusion responses against dynamic attackers," 2023.

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

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

[22] 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.

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

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

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

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

[27] 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.

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

[29] J. D. Day and H. Zimmermann, "The OSI reference model," *Proceedings of the IEEE*, vol. 71, no. 12, pp. 1334–1340, 1983.

[30] 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.

[31] S. Seuken and S. Zilberstein, "Formal models and algorithms for decentralized decision making under uncertainty," *Autonomous Agents and Multi-Agent Systems*, 2008.

[32] 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.

[33] 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.

[34] 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.

[35] 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.

[36] L. Huang, J. Chen, and Q. Zhu, *Factored Markov Game Theory for Secure Interdependent Infrastructure Networks*, 2018.

[37] F. A. Oliehoek, S. Whiteson, and M. T. J. Spaan, "Exploiting agent and type independence in collaborative graphical bayesian games," 2014.

[38] M. Kearns, M. Littman, and S. Singh, "Graphical models for game theory," 2013.

[39] 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.

[40] 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.

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. 1; each row contains the configuration of one or more components; vulnerabilities are identified according to the CVE and CWE databases [63], [64].

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. 1).

[41] 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.

[42] J. F. Nash, "Non-cooperative games," *Annals of Mathematics*, 1951.

[43] 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.

[44] 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.

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

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

[47] S. Banach, "Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales," *Fundamenta Mathematicae*, 1922.

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

[49] D. M. Topkis, "Minimizing a submodular function on a lattice," *Operations Research*, vol. 26, no. 2, pp. 305–321, 1978.

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

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

[52] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, 2017, http://arxiv.org/abs/1707.06347.

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

[54] 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.

[55] R. team, *RYU SDN Framework - English Edition*, ser. Release 1.0. RYU project team, 2014.

[56] B. Pfaff, J. Pettit, T. Koponen, E. Jackson, A. Zhou, J. Rajahalme, J. Gross, A. Wang, J. Stringer, P. Shelar, K. Amidon, and M. Casado, "The design and implementation of open vSwitch," in *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. Oakland, CA: USENIX Association, May 2015, pp. 117–130.

[57] N. McKeown *et al.*, "Openflow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, p. 69–74, mar 2008.

[58] M. Mahalingam *et al.*, "Virtual extensible local area network (vxlan): A framework for overlaying virtualized layer 2 networks over layer 3 networks," 2014, https://www.rfc-editor.org/rfc/rfc7348.

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

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

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

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

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

[64] ——, "Cwe list," 2023, https://cwe.mitre.org/index.html.

[65] 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.

[66] F. Timbers *et al.*, "Approximate exploitability: Learning a best response in large games," 2020, https://arxiv.org/abs/2004.09677.

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