Optimal Resource Allocation for Proactive Defense with Deception in Probabilistic Attack Graphs ⋆ ⋆⋆

Haoxiang Ma[1], Shuo Han[2], Charles Kamhoua[3], and Jie Fu[4]

[1] University of Florida, Gainesville FL 32611, USA hma2@ufl.edu
[2] University of Illinois Chicago, Chicago IL 60607, USA hanshuo@uic.edu
[3] DEVCOM Army Research Laboratory, Adelphi MD 20783, USA charles.a.kamhoua.civ@mail.mil
[4] University of Florida, Gainesville FL 32611, USA fujie@ufl.edu

Abstract. This paper investigates the problem of synthesizing proactive defense systems with deception. We model the interaction between the attacker and the system using a formal security model: a probabilistic attack graph. By allocating fake targets/decoys, the defender aims to distract the attacker from compromising true targets. By increasing the cost of some attack actions, the defender aims to discourage the attacker from committing to certain policies. To optimally deploy limited decoy resources and modify attack action costs with operational constraints, we formulate the synthesis problem as a bi-level optimization problem, while the defender designs the system, in anticipation of the attacker's best response given that the attacker has disinformation about the system due to the use of decoys. We investigate the bi-level optimization formulation against both rational and bounded rational attackers. We show the problem against a rational attacker can be formulated as a bi-level linear program. For attackers with bounded rationality, we show that under certain assumptions, the problem can be transformed into a constrained optimization problem. We proposed an algorithm to approximately solve this constrained optimization problem using a novel projected gradient ascent based on the idea of incentive-design. We demonstrate the effectiveness of the proposed methods using experiments and provide our insights in defense design against rational and bounded rational attackers.

Keywords: Deception · Attack Graph · Bi-Level Optimization · Markov Decision Process

1 Introduction

Proactive defense refers to a class of defense mechanisms for the defender to detect any ongoing attacks, distract the attacker with deception, or use randomization to increase the difficulty of an attack to the system. In this paper, we propose a mathematical framework and solution approach for synthesizing a proactive defense system with deception.

⋆ This work was sponsored in part by the Army Research Office and was accomplished under Grant Number W911NF-22-1-0034 and W911NF2210166 and in part by NSF under grant No. 2144113.

⋆⋆ DISTRIBUTION A: Approved for Public Release. Distribution is Unlimited.

We start by formulating the attack planning problem using a probabilistic attack graph, which can be viewed as a Markov decision process (MDP) with a set of attack target states. Attack graphs(AGs)[9] can be used in modeling computer networks. They are widely used in network security to identify the minimal subset of vulnerability/sensors to be used in order to prevent all known attacks[17,20]. Probabilistic attack graphs introduce uncertain outcomes of attack actions that account for action failures in a stochastic environment. For example, in [8,7], probabilistic transitions in attack graphs capture uncertainties originating from network-based randomization. Under the probabilistic attack graph modeling framework, we investigate how to allocate decoy resources as fake targets to distract the attacker into attacking the fake targets and how to modify the attack action costs to discourage the attacker from reaching the true targets.

The joint design of decoy resource allocation and action cost modification can be cast as a bi-level optimization problem, where the defender (at the upper level) designs the system, in anticipation of the attacker's (at the lower level) best response, given that the attacker has disinformation about the system due to allocated decoys. However, bi-level optimization problems are generally NP-hard [4]. We investigate two possible types of attackers: A rational attacker who maximizes the total reward and a bounded rational attacker whose action choices are computed using quantal response [3,12], where the probability of an action is proportional to the exponential of the total (discounted) return of that action.

For the rational attacker, we show that the bi-level optimization problem can be converted into a single-level optimization problem using Karush–Kuhn–Tucker (KKT) conditions of the lower-level optimization problem. For the bounded rational attacker, we formulated a constrained optimization problem and developed a new projected gradient ascent method to solve a (local) optimal policy. We build two important relations: First, we show that the projection step of a defender's desired attack policy to the set of realizable attack policy space can be performed using Inverse Reinforcement Learning (IRL) [24]. Essentially, IRL shapes the attacker's *perceived reward* so that the rational attacker will mimic a strategy chosen by the defender. Second, the gradient ascent step can be performed using policy improvement, which is a subroutine in policy iteration with respect to maximizing the defender's total reward. The projected gradient ascent is ensured to converge to a (local) optimal solution to this nonconvex-constrained optimization problem.

Related work The proactive defense design problem is closely related to the Stackelberg security game(SSG) (surveyed in [22]). In an SSG, the defender is to protect a set of targets with limited resources, while the attacker selects the optimal attack strategy given the knowledge of the defender's strategy. In [16], the authors study security countermeasure-allocation and use attack graphs to evaluate the network's security given the allocated resources. However, traditionally SSG does not account for the use of deception.

Deceptions create incorrect/incomplete information for the attacker. In [23], the authors formulate a security game to allocate limited decoy resources to mask a network configuration from the cyber attacker. The decoy-based deception manipulates the adversary's perception of the payoff matrix. In [2], the authors study honeypot allocation

in deterministic attack graphs and determine the optimal allocation strategy using the minimax theorem. In [13], the authors study directed acyclic attack graphs that can be modified by the defender using deceptive and protective resources. They propose a mixed-integer linear program (MILP)-based algorithm to determine the allocation of deceptive and protective resources in the graph. In [5], they harden the network by using honeypots so that the attacker can not discriminate between a true target and a fake target. In [14], the authors assign fake edges in the attack graph to interdict the attacker and employ MILP to find the optimal solution.

Compared to existing work, our work makes the following contributions: First, we do not assume any graph structure in the attack graph and consider probabilistic attack graphs instead of deterministic ones. As the attacker can take a randomized strategy in the probabilistic attack graph, it is impossible to construct a payoff matrix and apply the minimax theorem for decoy resource allocation. Second, we consider simultaneously allocating limited decoy resources and modifying the cost of attack actions, and analyzing the best response of the attacker given the disinformation caused by deception. Third, we propose tractable solutions for dealing with different types of attackers: rational and bounded rational. We show that by modifying the action reward and decoy resource allocation properly, it is possible to shape the attacker's behavior so that the misperceived attacker is incentivized to commit an attack strategy that maximizes the defender's reward. Finally, we evaluate our solution under different attacker types and test the scalability of our method on different problem sizes.

2 Preliminaries and Problem Formulation

Notations Let $\mathbf{R}$ denote the set of real numbers and $\mathbf{R}^n$ the set of real n-vectors. Let $\mathbf{R}^n_{>0}$ (resp. $\mathbf{R}^n_{<0}$) be the set of positive (resp. negative) real n-vectors. We use $\mathbf{1}$ to represent the vector of all ones. Given a vector $z \in \mathbf{R}^n$, let z_i be the i-th component. Given a finite set Z, the set of probability distributions over Z is represented as $\mathsf{Dist}(Z)$. Given $d \in \mathsf{Dist}(Z)$, the support of d is denoted as $\mathsf{Supp}(d) = \{z \in Z \mid d(z) > 0\}$. Let I_B be the indicator function, i.e., $I_B(x) = 1$ if $x \in B$, and $I_B(x) = 0$ otherwise.

We consider the adversarial interaction between a defender (player 1, pronoun she/her) and an attacker (player 2, pronoun he/him/his) in a system equipped with proactive defense (formally defined later). We first introduce a formal model, called probabilistic attack graph, to capture how the attacker plans to achieve the attack objective. Then, we introduce proactive defense countermeasures with deception.

Attack Planning Problem The attack planning problem is modeled as a probabilistic attack graph,

$$M = (S, A, P, \nu, \gamma, F, R_2),$$

where S is a set of states (nodes in the attack graph), A is a set of attack actions, $P : S \times A \to \mathsf{Dist}(S)$ is a probabilistic transition function such that $P(s'|s, a)$ is the probability of reaching state s' given action a being taken at state s, $\nu \in \mathsf{Dist}(S)$ is the initial state distribution, $\gamma \in (0, 1]$ is a discount factor. The attacker's objective is described by a set F of *target states* and a *target reward* function $R_2 : F \times A \to \mathbf{R}_{\geq 0}$, which assigns each state-action pair (s, a) where $s \in F$ and $a \in A$ to a nonnegative

value of reaching that target for the attacker. The reward function can be extended to the entire state space by defining $R_2(s,a) = 0$ for any $s \in S \setminus F, a \in A$. To capture the termination of attacks, we introduce a unique sink state $s_{\mathsf{sink}} \in S \setminus F$ such that $P(s_{\mathsf{sink}}|s_{\mathsf{sink}},a) = 1$ for all $a \in A$ and $P(s_{\mathsf{sink}}|s,a) = 1$ for any target $s \in F$ and $a \in A$.

The probabilistic attack graph characterizes goal-directed attacks encountered in cyber security [10,18], in which by reaching a target state, the attacker compromises certain critical network hosts. Probabilistic attack graphs [21,13] capture the uncertain outcomes of the attack actions using the probabilistic transition function and generalize deterministic attack graphs [9].

The attacker is to maximize his discounted total reward, starting from the initial state $S_0 \sim \nu$. A randomized, finite-memory attack policy is a function $\pi\colon S^* \to \mathsf{Dist}(A)$, which maps a finite run $\rho \in S^*$ into a distribution $\pi(\rho)$ over actions. A policy is called Markovian if it only depends on the most recent state, i.e., $\pi\colon S \to \mathsf{Dist}(A)$. We only consider Markovian policies because it suffices to search within Markovian policies for an optimal attack policy.

Let $(\Omega, \mathcal{F})$ be the canonical sample space for $(S_0, A_0, (S_t, A_t)_{t>1})$ with the Borel σ-algebra $\mathcal{F} = \mathcal{B}(\Omega)$ and $\Omega = S \times A \times \prod_{t=1}^{\infty}(S \times A)$. The probability measure $\mathbf{Pr}^\pi$ on $(\Omega, \mathcal{F})$ induced by a Markov policy π satisfies: $\mathbf{Pr}^\pi(S_0 = s) = \mu_0(s)$, $\mathbf{Pr}^\pi(A_0 = a \mid S_0 = s) = \pi(s,a)$, and $\mathbf{Pr}^\pi(S_t = s \mid (S_k, A_k)_{k<t}) = P(s \mid S_k, A_k)$, and $\mathbf{Pr}^\pi(A_t = a \mid (S_k, A_k)_{k<t}, S_t) = \pi(S_t, a)$.

Given a Markovian policy $\pi\colon S \to \mathsf{Dist}(A)$, we define the attacker's value function $V_2^\pi : S \to \mathbf{R}$ as $V_2^\pi(s) = \mathbf{E}_\pi[\sum_{k=0}^{\infty} \gamma^k R_2(S_k, A_k) | S_0 = s]$, where $\mathbf{E}_\pi$ is the expectation given the probability measure $\mathbf{Pr}^\pi$.

Proactive Defense with Deception We assume that the defender knows the attacker's objective given by the tuple $\langle F, R_2 \rangle$, i.e., the target states and target reward function. The defender's proactive defense mechanisms are the following:

- Defend by deception: The defender employs a deception method called "revealing the fake". Specifically, the defender has a set $D \subset S \setminus F$ of states in the MDP M that can be set to be *fake target states* with fake target rewards $\boldsymbol{y} \in \mathbf{R}^{|D|}$. The attacker cannot distinguish the real targets F from fake targets D.
- Defend by state-action reward modification: The defender has a set $W \subset (S \setminus (F \cup D)) \times A$ of state action pairs in the MDP M whose reward can be modified. Once the reward of the state action pair (s,a) is modified, the attacker's perceived reward $R_2(s,a) < 0$, i.e., the cost of attack action a at state s is $-R_2(s,a)$.

The defender can determine how to allocate her decoy resource and limited state-action reward modification ability.

Definition 1 (Decoy allocation under constraints). *The defender's decoy allocation design is a nonnegative real-valued vector $\boldsymbol{y} \in \mathbf{R}_{\geq 0}^{|S|}$ satisfying $\boldsymbol{y}(s) = 0$ for any $s \in S \setminus D$ and constrained by $\mathbf{1}^\mathsf{T}\boldsymbol{y} \leq h$ for some $h \geq 0$. Given a decoy allocation $\boldsymbol{y}$,*

the attacker's perceptual reward function *is defined by*

$$R_2^{\boldsymbol{y}}(s,a) = \begin{cases} \boldsymbol{y}(s) & \text{if } \boldsymbol{y}(s) > 0, \\ R_2(s,a) & \text{if } \boldsymbol{y}(s) = 0. \end{cases}$$

Definition 2 (Action reward modification). *Given a set $W \subset (S \setminus (F \cup D)) \times A$, the defender's action reward modification is a nonpositive reward-valued vector $\boldsymbol{x} \in \mathbf{R}_{\leq 0}^{|S \times A|}$ satisfying $\boldsymbol{x}(s,a) = 0$ for any $(s,a) \notin W$ and $-\mathbf{1}^{\mathsf{T}}\boldsymbol{x} \leq k$ for some $k \geq 0$. Given an action reward modification $\boldsymbol{x}$, the attacker's perceptual reward function is defined by*

$$R_2^{\boldsymbol{x}}(s,a) = \begin{cases} \boldsymbol{x}(s,a) & \text{if } \boldsymbol{x}(s,a) < 0, \\ R_2(s,a) & \text{if } \boldsymbol{x}(s,a) = 0. \end{cases}$$

The defender does not consider modifying the state-action reward for (fake or real) target states $F \cup D$ because once a state in $F \cup D$ is reached, the attack is terminated.

Definition 3. *The defender's* proactive defense strategy *is a tuple $(\boldsymbol{x}, \boldsymbol{y})$ including an action reward modification $\boldsymbol{x}$ and a decoy allocation design $\boldsymbol{y}$.*

Because the action reward modification is independent of the decoy allocation design, the reward function given a defender's strategy $(\boldsymbol{x}, \boldsymbol{y})$ is the composition of $R_2^{\boldsymbol{x}}$ and $R_2^{\boldsymbol{y}}$ and thus omitted.

Assumption 1 *The attack process terminates under two cases: Either the attack succeeds, in which the attacker reaches a target $s \in F$, or the attack is interdicted, in which the attacker reaches a state allocated with a decoy.*

Our problem can be informally stated as follows.

Problem 1. In the attack planning scenario we mentioned above, determine the defender's strategy to allocate decoy resources and modify action rewards so as to maximize the probability that the attacker reaches a fake target given the best response of the attacker.

3 Main Results

In this section, we first define the attacker's perceptual planning problem for a fixed action reward modification and decoy resource allocation $(\boldsymbol{x}, \boldsymbol{y})$. Then we show that the design of the proactive defense can be formulated as a bi-level optimization problem. We investigate the special property of the formulated bi-level optimization problem to develop an optimization-based approach for synthesizing the proactive defense strategy.

3.1 A Bi-level Optimization Formulation

The defender's strategy changes how the attacker perceives the attack planning problem as follows:

Definition 4 (Perceptual attack planning problem with modified reward and decoys). *Let the action reward modification be $\boldsymbol{x}$ and decoy allocation be $\boldsymbol{y}$, and the attacker's original planning problem $M = (S, A, P, \nu, \gamma, F, R_2)$, the perceptual planning problem of the attacker is defined by the following MDP with terminating states:*

$$M(\boldsymbol{x}, \boldsymbol{y}) = (S, A, P^{\boldsymbol{y}}, \nu, \gamma, F \cup D^{\boldsymbol{y}}, R_2^{\boldsymbol{x},\boldsymbol{y}}),$$

where S, A, ν, γ are the same as those in M, $D^{\boldsymbol{y}} = \{s \in D \mid \boldsymbol{y}(s) \neq 0\}$ are decoy target states and absorbing. The transition function $P^{\boldsymbol{y}}$ is obtained from the original transition function P by only making all states in $D^{\boldsymbol{y}}$ absorbing. The reward $R_2^{\boldsymbol{x},\boldsymbol{y}}$ is defined based on Def. 1 and Def. 2.

The perceptual value for the attacker is

$$V_2^{\pi}(\nu; \boldsymbol{x}, \boldsymbol{y}) = \mathbf{E}_{\pi}\Big[\sum_{k=0}^{\infty} \gamma^k R_2^{\boldsymbol{x},\boldsymbol{y}}(S_k, A_k) \mid S_0 \sim \nu\Big],$$

where $\mathbf{E}_{\pi}$ is the expectation given the probability measure $\mathbf{Pr}^{\pi}$ induced by π from the MDP $M(\boldsymbol{x}, \boldsymbol{y})$.

The defender's deception objective is given by a reward function $R_1^{\boldsymbol{y}} : S \to \mathbf{R}$, defined by

$$R_1^{\boldsymbol{y}}(s) = \begin{cases} 1 & \text{if } s \in D^{\boldsymbol{y}}, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

Given the probability measure $\mathbf{Pr}^{\pi}$, we denote the defender's value by

$$V_1^{\pi}(\nu; \boldsymbol{y}) = \mathbf{E}_{\pi}\Big[\sum_{k=0}^{\infty} \gamma^k R_1(S_k) \mid S_0 \sim \nu\Big].$$

With this reward definition, the value $V_1^{\pi}(\nu; \boldsymbol{y})$ is the probability of the attacker reaching a fake target in $D^{\boldsymbol{y}}$.

Then the problem of synthesizing an optimal proactive defense strategy $(\boldsymbol{x}, \boldsymbol{y})$ can be mathematically formulated as

Problem 2.

$$\begin{aligned} \max_{\boldsymbol{x} \in X, \boldsymbol{y} \in Y} \quad & V_1^{\pi^*}(\nu; \boldsymbol{y}) \\ \text{s.t.} \quad & \pi^* \in \operatorname*{argmax}_{\pi} V_2^{\pi}(\nu; \boldsymbol{x}, \boldsymbol{y}). \end{aligned}$$

where $X = \{\boldsymbol{x} \in \mathbf{R}_{\leq 0}^{|W|} \mid -\mathbf{1}^{\mathsf{T}}\boldsymbol{x} \leq k\}$ and $Y = \{\boldsymbol{y} \mid \forall s \in S \backslash D, \boldsymbol{y}(s) = 0 \text{ and } \mathbf{1}^{\mathsf{T}}\boldsymbol{y} \leq h\}$ are the ranges for variables $\boldsymbol{x}$ and $\boldsymbol{y}$ correspondingly.

In words, the defender decides $(\boldsymbol{x}, \boldsymbol{y})$ so that the attacker's best response in his perceptual attack planning problem turns out to be an attack policy most preferred by the defender, as it maximizes the defender's value.

3.2 Synthesizing proactive defense against a rational attacker

The bi-level optimization problem is known to be strongly NP-hard [6]. In this section, we show that when the attacker is rational, then the lower-level problem can be formulated as a linear program (LP). Thus, the original bi-level optimization is a special case–bi-level LP. Using the KKT condition of the lower-level problem, the bi-level LP reduces to a single-level optimization with special ordered set(SOS) constraints. We formulate the lower-level LP using occupancy measures [1]. For a given defense strategy $(\boldsymbol{x}, \boldsymbol{y})$, the optimal policy perceived by the attacker can be solved using the following LP:

$$\begin{aligned}
&\max_{m} \sum_{s\in S, a\in A} R_2^{\boldsymbol{x},\boldsymbol{y}}(s,a)m(s,a).\\
&\text{s.t.} \quad \sum_{a\in A} m(s,a) = \gamma \sum_{s'\in S, a'\in A} P(s|s',a')m(s',a') + \nu(s), \forall s \in S, \qquad (2)\\
&\qquad m(s,a) \geq 0, \forall s \in S, a \in A. \qquad (3)
\end{aligned}$$

where $m(s,a)$ is the (discounted) occupancy measure that represents the frequency a state s is visited and a is taken. Using the solution of the LP, the optimal attacker policy π is recovered via: $\pi(s,a) = \frac{m(s,a)}{\sum_{a'\in A} m(s,a')}$.

The original bi-level optimization reduces to

$$\begin{aligned}
&\max_{\boldsymbol{x}\in X, \boldsymbol{y}\in Y} \quad \sum_{s\in S, a\in A} R_1(s,a)m(s,a)\\
&\text{s.t.} \quad \max_{m} \sum_{s\in S, a\in A} R_2^{\boldsymbol{x},\boldsymbol{y}}(s,a)m(s,a), \quad \text{s.t. (2), (3).}
\end{aligned}$$

By rewriting the lower-level LP using its KKT conditions, we convert the bi-level optimization problem into a single-level optimization problem with SOS1 constraints.

First, we have the lower-level problem:

$$\begin{aligned}
&\max_{m} \sum_{s\in S, a\in A} R_2^{\boldsymbol{x},\boldsymbol{y}}(s,a)m(s,a).\\
&\text{s.t.} \quad \sum_{a\in A} m(s,a) = \gamma \sum_{s'\in S, a'\in A} P(s|s',a')m(s',a') + \nu(s), \forall s \in S, \qquad (4)\\
&\qquad m(s,a) \geq 0, \forall s \in S, a \in A. \qquad (5)
\end{aligned}$$

where we have $R_2^{\boldsymbol{x},\boldsymbol{y}}(s,a) = R_2(s,a) + \boldsymbol{x}(s,a) + \boldsymbol{y}(s), \forall s \in S, a \in A$. Thus we can use KKT condition to form the lower-level problem to a Lagrangian function. We first rewrite

$$\sum_{a\in A} m(s,a) = \gamma \sum_{s'\in S, a'\in A} P(s|s',a')m(s',a') + \nu(s), \forall s \in S. \qquad (6)$$

to the matrix form, which is equivalent to

$$\mathcal{C}\boldsymbol{m} - \gamma\mathcal{D}\boldsymbol{m} - \nu = 0.$$

where $\mathcal{C}, \mathcal{D}$ corresponds to the parameters in Equation 6. And $\boldsymbol{m} \in \mathbf{R}_{\geq 0}^{|S\times A|}$ denotes the vector of discounted state-action visiting frequency.

Thus the Lagrangian function can be written as

$$\mathcal{L}(\boldsymbol{m}, \mu, \lambda) = (\boldsymbol{R_2} + \boldsymbol{x} + \boldsymbol{y})^T\boldsymbol{m} + \mu^T\boldsymbol{m} + \lambda^T(\mathcal{C}\boldsymbol{m} - \gamma\mathcal{D}\boldsymbol{m} - \nu). \tag{7}$$

where $\boldsymbol{y}$ is extended to $S \times A$ domain by defining $\boldsymbol{y}(s, a) = \boldsymbol{y}(s)$, and $\boldsymbol{R_2}$ is the vector form of reward function R_2.

The necessary conditions are listed as follows:

$$\begin{aligned}
&-(\boldsymbol{R_2} + \boldsymbol{x} + \boldsymbol{y}) + \mu + (\mathcal{C} - \gamma\mathcal{D})^T\lambda = \mathbf{0},\\
&\mathcal{C}\boldsymbol{m} - \gamma\mathcal{D}\boldsymbol{m} - \nu = 0,\\
&-\boldsymbol{m} \leq \mathbf{0},\\
&\mu \geq \mathbf{0},\\
&\mu(i)\boldsymbol{m}(i) = 0, i = 1, 2, ..., |S \times A|.
\end{aligned} \tag{8}$$

where (8) are special ordered sets of type 1 (SOS1) constraints. We then combine these necessary conditions with the upper-level problem, the bi-level problem can be rewritten as:

$$\begin{aligned}
\max_{\boldsymbol{x}\in X, \boldsymbol{y}\in Y, \boldsymbol{m}, \mu, \lambda} \quad & \boldsymbol{R_1}^T\boldsymbol{m}\\
\text{s.t.} \quad & \mathbf{1}^\mathsf{T}\boldsymbol{y} \leq h,\\
& -\mathbf{1}^\mathsf{T}\boldsymbol{x} \leq k,\\
& \boldsymbol{y} \geq 0,\\
& \boldsymbol{x} \leq 0,\\
& -(\boldsymbol{R_2} + \boldsymbol{x} + \boldsymbol{y}) + \mu + (\mathcal{C} - \gamma\mathcal{D})^T\lambda = \mathbf{0},\\
& \mathcal{C}\boldsymbol{m} - \gamma\mathcal{D}\boldsymbol{m} - \nu = 0,\\
& -\boldsymbol{m} \leq \mathbf{0},\\
& \mu \geq \mathbf{0},\\
& \mu(i)\boldsymbol{m}(i) = 0, i = 1, 2, ..., |S \times A|.
\end{aligned} \tag{9}$$

where $\boldsymbol{R_1}$ is the vector form of reward function R_1. This optimization problem is single-level and can be solved using the Gurobi Optimization toolbox.

3.3 Synthesizing proactive defense against a bounded rational attacker

The defense against a rational agent can be sensitive to potential mismatches on the rationality assumption: Consider the defender aims to protect two targets $\{1, 2\}$. Both

targets have similar values but target 1's value is slightly higher than that of target 2. Knowing a rational agent will aim at target 1, the defender will enforce all resources to guard target 1 and may leave target 2 unprotected. However, a bounded rational attacker, based on the quantal response [3,12], will compromise either target with almost equal probabilities. We investigate how to design a defense strategy against attackers with bounded rationality.

Transforming into a Constrained Optimization Problem Based on the quantal response model, an attacker with bounded rationality aims to compute a quantal response policy π^* in the perceived MDP $M(\boldsymbol{x}, \boldsymbol{y})$ by solving the following entropy-regularized Bellman equation [15]:

$$V_2^*(s; \boldsymbol{x}, \boldsymbol{y}) = \tau \log \sum_a \exp\{(R_2(s; \boldsymbol{x}, \boldsymbol{y}) + \gamma V_2^*(s; \boldsymbol{x}, \boldsymbol{y}))/\tau\},$$

where $\tau > 0$ is the temperature parameter that controls the degree of entropy regularization, if τ approaches 0, the Bellman equation recovers the optimal Bellman equation under a rational attacker. However, due to the bounded rationality assumption, the original bi-level optimization cannot be reduced into a bi-level LP as the objective function using occupancy measures includes an additional nonlinear term which is the weighted entropy of the policy.

Next, we propose a gradient-based method to solve Problem 2 assuming the attacker is bounded rational. First, we show the original problem can be formulated as a constrained optimization problem. Let $\mathit{\Pi}(\boldsymbol{x}, \boldsymbol{y})$ be the set of quantal response policies in the attacker's perceived planning problem with respect to a choice of variables $\boldsymbol{x}$ and $\boldsymbol{y}$. The bi-level optimization problem is then equivalently written as the following constrained optimization problem:

$$\begin{aligned} \max_{\pi^*, \boldsymbol{x} \in X, \boldsymbol{y} \in Y} \quad & V_1^{\pi^*}(\nu; \boldsymbol{y}) \\ \text{s.t.} \quad & \pi^* \in \mathit{\Pi}(\boldsymbol{x}, \boldsymbol{y}). \end{aligned} \tag{10}$$

This, in turn, is equivalent to

$$\begin{aligned} \max_{\pi^*} \quad & V_1^{\pi^*}(\nu; \boldsymbol{y}) \\ \text{s.t.} \quad & \pi^* \in \bigcup_{\boldsymbol{x} \in X, \boldsymbol{y} \in Y} \mathit{\Pi}(\boldsymbol{x}, \boldsymbol{y}). \end{aligned} \tag{11}$$

Here, the constraint means the attacker's response π^* can be selected from the collection of optimal attack policies given all possible values for $\boldsymbol{x}$, $\boldsymbol{y}$.

By the definition of the defender's value function, it is noted that $V_1^{\pi}(\nu; \boldsymbol{y})$ does not depend on the exact value of $\boldsymbol{y}$ but only depends on whether $\boldsymbol{y}(s) > 0$ for each state $s \in D$. Formally,

Lemma 1. *For any $\boldsymbol{y}_1, \boldsymbol{y}_2 \in Y$, if $\boldsymbol{y}_1(s) = 0 \implies \boldsymbol{y}_2(s) = 0$ and vice versa, then $V_1^{\pi}(\nu; \boldsymbol{y}_1) = V_1^{\pi}(\nu; \boldsymbol{y}_2)$.*

Proof. Given two different vectors $\boldsymbol{y_1}$ and $\boldsymbol{y_2}$, we can construct two MDPs: $M_1 \coloneqq M(\boldsymbol{x}, \boldsymbol{y_1}) = (S, A, P^{\boldsymbol{y_1}}, \nu, \gamma, F, R_1)$ and $M_2 \coloneqq M(\boldsymbol{x}, \boldsymbol{y_2}) = (S, A, P^{\boldsymbol{y_2}}, \nu, \gamma, F, R_1)$, respectively.

If $\boldsymbol{y_1}(s) = 0$ if and only if $\boldsymbol{y_2}(s) = 0$, then the transition functions $P^{\boldsymbol{y}_1}$ of M_1 and $P^{\boldsymbol{y}_2}$ of M_2 are the same (see Def. 4).

Further, the defender's reward function $R_1^{\boldsymbol{y}_1}$ also equals to $R_1^{\boldsymbol{y}_2}$ (see (1)), given both the transition dynamics and reward are the same, we have $V_1^{\pi}(\nu; \boldsymbol{y}_1) = V_1^{\pi}(\nu; \boldsymbol{y}_2)$.

Lemma 1 proves given an attacker's policy π, the defender's value only relates to where the decoys are located. Next, to remove the dependency of $V_1^{\pi}(\nu; \boldsymbol{y})$ on $\boldsymbol{y}$, we make the following assumption:

Assumption 2 *The set $D^{\boldsymbol{y}} = \{s \in D \mid \boldsymbol{y}(s) \neq 0\}$ of states where decoys are allocated is given.*

Under this assumption, we simply assume all states in the given set D have to be assigned with nonzero decoy resources. That is $D^{\boldsymbol{y}} = D$.

This assumption further reduces the defender's synthesis problem into a constrained optimization problem.

$$\begin{aligned} \max_{\pi^*} \quad & V_1^{\pi^*}(\nu) \\ \text{s.t.} \quad & \pi^* \in \overline{\Pi} \triangleq \bigcup_{\boldsymbol{y} \in Y, \boldsymbol{x} \in X} \Pi(\boldsymbol{x}, \boldsymbol{y}), \\ & \boldsymbol{y}(s) > 0, \forall s \in D. \end{aligned} \tag{12}$$

Because the above problem is a standard-constrained optimization problem, one can obtain a locally optimal solution using the projected gradient method:

$$\pi^{k+1} = \mathrm{proj}_{\overline{\mathbf{\Pi}}}(\pi^k + \eta \nabla V_1^{\pi^k}(\nu)).$$

where $\mathrm{proj}_{\overline{\mathbf{\Pi}}}(\pi)$ denotes projecting policy π onto the policy space $\overline{\Pi}$ and η is the step size.

Connecting Inverse-reinforcement Learning with Projected Gradient Ascent A key step in performing Projected Gradient Ascent (PGA) is to evaluate, for any policy $\hat{\pi}$, the projection $\mathrm{proj}_{\overline{\mathbf{\Pi}}}(\hat{\pi})$. However, this is nontrivial because the set $\bar{\Pi}$ includes a set of attack policies, each of which corresponds to a choice of vectors $(\boldsymbol{x}, \boldsymbol{y})$. As a result, $\bar{\Pi}$ does not have a compact representation. Next, we propose a novel algorithm that computes the projection.

First, it is noted that this projection step is equivalent to solving the following optimization problem:

$$\begin{aligned} \min_{\pi} \quad & \mathbf{D}(\hat{\pi}, \pi) \\ \text{s.t.} \quad & \pi \in \overline{\Pi}, \\ & \boldsymbol{y}(s) > 0; \forall s \in D. \end{aligned} \tag{13}$$

where $\mathbf{D}(\hat{\pi}, \pi)$ is the distance between the two policies $\hat{\pi}, \pi$.

The distance function $\mathbf{D}$ can be chosen to be the Kullback–Leibler (KL)-divergence between policy-induced Markov chains. Specifically, the KL divergence in (13) can be expressed as

$$\begin{aligned}\mathbf{D}_{\mathrm{KL}}\left(M_{\widehat{\pi}}(\boldsymbol{x},\boldsymbol{y})\|M_{\pi}(\boldsymbol{x},\boldsymbol{y})\right) &= \sum_{\rho}\widehat{\mathbf{Pr}}(\rho)\log\frac{\widehat{\mathbf{Pr}}(\rho)}{\mathbf{Pr}(\rho|\boldsymbol{x},\boldsymbol{y})}\\ &= \sum_{\rho}\widehat{\mathbf{Pr}}(\rho)\log\widehat{\mathbf{Pr}}(\rho) - \sum_{\rho}\widehat{\mathbf{Pr}}(\rho)\log\mathbf{Pr}(\rho|\boldsymbol{x},\boldsymbol{y}), \quad (14)\end{aligned}$$

where $\widehat{\mathbf{Pr}}(\rho)$ is the probability of path ρ in the Markov chain $M_{\widehat{\pi}}(\boldsymbol{x},\boldsymbol{y})$, and $\mathbf{Pr}(\rho|\boldsymbol{y})$ is the probability of path ρ in the Markov chain $M_{\pi}(\boldsymbol{x},\boldsymbol{y})$ induced by a policy π.

Because the first term in the sum in (14) is a constant for $\hat{\pi}$ is fixed, the KL divergence minimization problem is equivalent to the following maximization problem:

$$\max_{\boldsymbol{x}\in X,\boldsymbol{y}\in Y} \quad \sum_{\rho}\widehat{\mathbf{Pr}}(\rho)\log\mathbf{Pr}(\rho|\boldsymbol{x},\boldsymbol{y}) \quad (15)$$

Problem (15) can be solved by an extension of the Maximum Entropy (MAXENT) IRL algorithm [24], which was originally developed in the absence of constraints on the reward parameters. It is well-known that IRL is to infer, from the expert demonstrations, a reward function for which the policy generating the demonstrations is optimal.

The use of IRL to perform the projection is intuitively understood as follows: The goal is to compute a pair of vectors $(\boldsymbol{x},\boldsymbol{y})$ that alters the attacker's perceived reward function so that the bounded rational attacker's optimal policy given $(\boldsymbol{x},\boldsymbol{y})$ is closed to the "expert policy" $\hat{\pi}$, under the constraints of $\boldsymbol{x},\boldsymbol{y}$. Importantly, we used the MAXENT IRL because it assumes the expect policy is entropy-regulated, and thus is consistent with the assumption of the quantal response of a bounded rational attacker.

To enforce the constraints $\boldsymbol{x} \in X, \boldsymbol{y} \in Y$, we approximate the constraint using a logarithmic barrier function and compute the optimal solution $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ using gradient-based numerical optimization Considering the constraint $\mathbf{1}^{\mathsf{T}}\boldsymbol{y} \leq h$, we implement the barrier function to approximate the inequality constraints and rewrite the optimization problem as:

$$\begin{aligned}&\max_{\boldsymbol{x},\boldsymbol{y}} \sum_{\rho}\widehat{\mathbf{Pr}}(\rho)\log\mathbf{Pr}(\rho|\boldsymbol{x},\boldsymbol{y}) + \frac{1}{t}\log(h-\mathbf{1}^{\mathsf{T}}\boldsymbol{y}) + \frac{1}{t}\log(k+\mathbf{1}^{\mathsf{T}}\boldsymbol{x})\\ &\text{s.t.} \quad \boldsymbol{y}(s) = 0, \quad \forall s \in S \setminus D,\\ &\qquad \boldsymbol{x}(s,a) = 0, \forall (s,a) \in S \times A \setminus W.\end{aligned}$$

where t is the weighting parameter of the logarithmic barrier function. In our experiment, t is fixed to be 1000.

Let $L(\boldsymbol{x},\boldsymbol{y})$ be the objective function. Specifically, $\boldsymbol{x}$ and $\boldsymbol{y}$ can be updated via $\boldsymbol{x}^{k+1} = \mathrm{proj}_{\mathbf{X}}\left(\boldsymbol{x}^k + \eta_x \nabla L(\boldsymbol{x},\boldsymbol{y})\right)$, $\boldsymbol{y}^{k+1} = \mathrm{proj}_{\mathbf{Y}}\left(\boldsymbol{y}^k + \eta_y \nabla L(\boldsymbol{x},\boldsymbol{y})\right)$.

Policy Improvement for Gradient Ascent Step After the projection step to obtain a policy π^k and the corresponding vector $(\boldsymbol{x}, \boldsymbol{y})$, we aim to compute a one-step gradient ascent to improve the objective function's value

$$V_1^{k+1}(\nu) = V_1^k(\nu) + \nabla V_1^k(\nu),$$

where $V_1^k(\nu)$ is the defender's value evaluated given the attack policy π^k at the k-th iteration.

For this step, we perform a policy improvement step with respect to the defender's reward function $R_1^{\boldsymbol{y}}$. It is shown in [19,11] that policy improvement is a one-step Newton update of optimizing the value function.

Specifically, the policy improvement is to compute

$$\tilde{\pi}^{k+1}(s,a) = \frac{\exp\left((R_1(s,a) + \gamma V_1^k(s'))/\tau\right)}{\sum_{a \in A} \exp\left((R_1(s,a) + \gamma V_1^k(s'))/\tau\right)},$$

The policy at iteration $k+1$ is obtained by performing the projection step ((13)) in which $\hat{\pi} \triangleq \tilde{\pi}_{k+1}$.

The iteration stops when $|V_1^{k+1}(\nu) - V_1^k(\nu)| \leq \epsilon$ where ϵ is a manually defined threshold. The output yields a tuple $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ which is the (local) optimal proactive defense strategy. We can only obtain a local optimal proactive defense strategy here due to the transferred constrained optimization problem having a nonconvex constraint set. However, we can start from different initial policies and select the best one.

To summarize, the proposed algorithm starts with an initial policy $\tilde{\pi}^0$, and use the IRL to find the projection π^0 as well as the corresponding vectors $(\boldsymbol{x}^0, \boldsymbol{y}^0)$ that shape the attacker's perceptual reward function for which π^0 is optimal. Then a policy improvement is performed to update π^0 to $\tilde{\pi}^1$. By alternating the projection and policy improvement, the process terminates until the stopping criteria is satisfied.

Remark 1. In our problem, we assume the set D is given. If the set D is not given but to be determined from a candidate set of states. Then the bi-level optimization is combinatorial and NP-hard. A naive approach is to enumerate all possible combinations and evaluate the defender's value for every subset and select the one that yields the highest defender's value.

4 Experiment

We illustrate the proposed methods with two sets of examples, one is a probabilistic attack graph and another is an attack planning problem formulated in a stochastic grid-world. For all case studies, the workstation used is powered by Intel i7-11700K and 32GB RAM.

Figure 1 shows a probabilistic attack graph with the targets $F = \{10\}$. The attacker has four actions $\{a, b, c, d\}$. For clarity, the graph only shows the transition given action a where a thick (resp. thin) arrow represents a high (resp. low) transition probability. For example, $P(0, a) = \{1 : 0.7, 2 : 0.1, 3 : 0.1, 4 : 0.1\}$ [5]. The defender can allocate

[5] The exact transition function is provided: https://www.dropbox.com/s/nyycf57vdry139j/MDPTransition.pdf?dl=0.

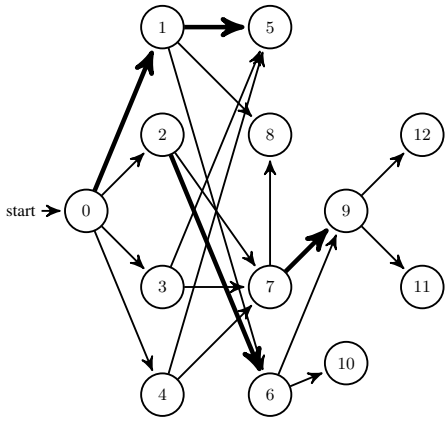

Fig. 1: A probabilistic attack graph.

decoy resources at a set $D = \{11, 12\}$ of decoy states and receive a reward of 1 if the attacker reaches the decoy instead of the true targets. If no decoy resource is allocated, the attacker receives a reward $R_2(s, a) = 1$ for any $s \in F$ and the optimal attack policy has probability 60.33% of reaching the target set F from the initial state 0. In the meantime, the defender's expected value is 0.149. That is, with probability 14.9%, the attacker will reach decoy set D and the attack is terminated.

Consider a defender who has a limited decoy resource constrained by $\mathbf{1}^\mathsf{T}\boldsymbol{y} \leq 3$ and cannot modify the state-action reward. First, we consider the decoy allocation against a rational attacker, from the bi-level LP solution, the decoy resource allocation is $\boldsymbol{y}_1(11) = 1.218, \boldsymbol{y}_1(12) = 0$. The defender's value is 0.654 given the best response of a rational attacker in $M(\boldsymbol{y}_1)$. Then, the same problem is solved for defending against a bounded rational attacker. The decoy resource allocation based on the PGA method yields $\boldsymbol{y}_2(11) = \boldsymbol{y}_2(12) = 1.313$. Based on the given decoy resource allocation, the attacker has an 8.63% probability of reaching the target set F and the defender's expected value is 0.653 at initial state 0. In these two cases, we observed that the defender's values are similar: By assigning resources to decoys to attract the attacker, the defender reduces the attacker's probability of reaching the target state significantly (85% reduction) and improves the defender's value by 3.38 times.

A key observation is that the decoy allocation against rational attacker $\boldsymbol{y}_1$ places resources only at one decoy state. This is because, when $\boldsymbol{y}_1(11) = 1.218$, the rational attacker selects the optimal action to reach state 9 and then 11 from state 6 instead of the true target 10. If the attacker is bounded rational, then at state 6, he will choose the action leading to either 9 or 10 with nearly equal probabilities. Thus, the design $\boldsymbol{y}_1$ against a rational attacker can be sensitive to possible mismatches in the rationality assumption. To see this, we perform the following comparison: We use the design $\boldsymbol{y}_1$ against a rational attacker to construct the attack planning MDP and then solve the optimal attack policy of a bounded rational attacker in this MDP. The defender's value is obtained by evaluating the bounded rational attacker policy in $M(\boldsymbol{y}_1)$ with the defender's reward. In this example, we observe that the defender's value is 0.444, which

indicates that the defender would have a performance drop of 33% if the rationality assumption is violated. On the other hand, when we solve a rational attack policy in the MDP $M(\boldsymbol{y}_2)$, whose defense is optimized against the bounded rational attacker, we observe the defender's value is 0.654, which is similar to the case against a bounded rational attacker. The result is shown in Table 1.

Defense Strategies \ Types of Attackers	Rational	Bounded Rational
$\boldsymbol{y}_1$ optimized for rational attackers	0.654	0.444
$\boldsymbol{y}_2$ optimized for bounded rational attackers	0.654	0.653

Table 1: Defender's values in the probabilistic attack graph.

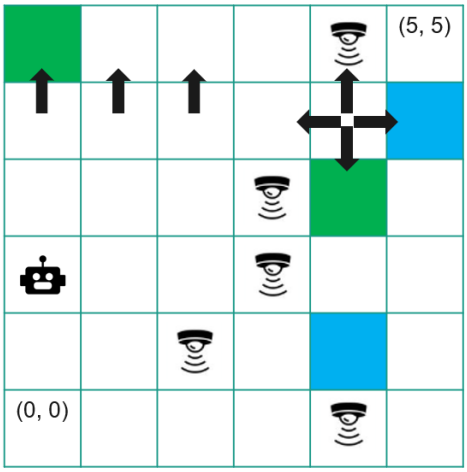

Fig. 2: A 6×6 gridworld.

Next, we consider a robot motion planning problem in attack graphs modeled by stochastic gridworlds. The purpose of choosing such an environment is to make the results more interpretable. Consider first a small 6 by 6 gridworld in Fig. 2. The attacker/robot aims to reach a set of goal states while avoiding detection from the defender. The attacker can move in four compass directions. Given an action, say, "N", the attacker enters the intended cell with $1 - 2\alpha$ probability and enters the neighboring cells, which are west and east cells with α probability. In our experiments, α is selected to be 0.1. A state (i, j) means the cell at row i and column j.

The defender has deployed sensors shown in Fig. 2 to detect an attack. Once the attacker enters a sensor state, his task fails. The decoy set D is given as blue cells and the target set F is given as green cells. The robot icon represents the robot's initial state. If no decoy resource is allocated, the attacker's policy has a probability of 98.98% of

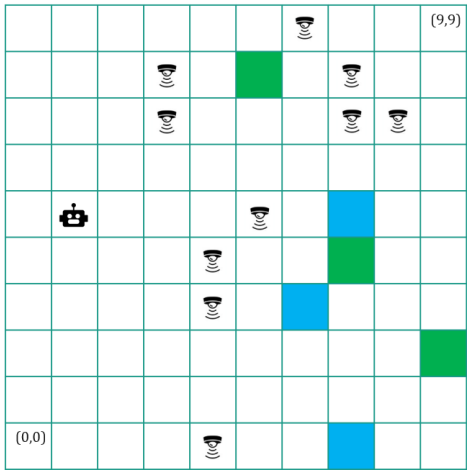

Fig. 3: A 10×10 gridworld.

reaching the target set from the initial state. In the meantime, the defender's expected value is 3.56×10^{-6}, which means the attacker's probability of reaching decoys is close to 0.

We employ the bi-level LP to solve decoy allocation against a rational attacker and the result is $\boldsymbol{y}_1((1,4)) = 1.946, \boldsymbol{y}_2((4,5)) = 1.774$, the defender's value is 0.433. Then the same problem is solved for defending against a bounded rational attacker. The PGA method yields $\boldsymbol{y}_2((1,4)) = 2.016, \boldsymbol{y}_2((4,5)) = 1.826$. Based on the given decoy resource allocation, the attacker has a 9.9% probability of reaching the target set F, and the defender's expected value at the initial state is 0.388.

To see how sensitive $\boldsymbol{y}_1$ is to the rationality assumption of the attacker, we evaluate the defense strategy $\boldsymbol{y}_1$, $\boldsymbol{y}_2$ against these two types of attackers: rational and bounded rational attackers. We observe a 26% decrease of defender's value when using $\boldsymbol{y}_1$, optimized against rational attacker, to defend against a bounded rational attacker. When the optimal defense $\boldsymbol{y}_2$ against a bounded rational attacker is used against a rational attacker, the performance loss for the defender is negligible. The result is shown in Table. 2.

The effects of allowing action-reward modifcation and different choices of decoy states We study how much the defense can be improved by allowing additional state-action reward modification. The actions the defender can modify are marked as arrows in Fig 2. The PGA method yields $\boldsymbol{x}_2((4,0), \text{'N'}) = -1, \boldsymbol{x}_2((4,1), \text{'N'}) = -0.94, \boldsymbol{x}_2((4,2), \text{'N'}) = -0.904, \boldsymbol{x}_2((4,4), \text{'N'}) = \boldsymbol{x}_2((4,4), \text{'W'}) = \boldsymbol{x}_2((4,4), \text{'S'}) = -1, \boldsymbol{x}_2((4,4), \text{'E'}) = 0, \boldsymbol{y}_2((1,4)) = 1.938, \boldsymbol{y}_2((4,5)) = 1.734$. The defender's value is 0.394 given the joint decoy allocation and action reward modification, and the attacker has a probability of 8.6% to reach the true goal, which is 13.13% reduction compared to that when only the decoy resource allocation is allowed. The result is shown in Table 3.

In order to test how the decoy set D influences the result. We re-allocate the position of decoys to $\{(0,2),(5,3)\}$. The result is shown in Table 4. If we do not allocate

Types of Attackers / Defense Strategies	Rational	Bounded Rational
$\boldsymbol{y}_1$ optimized for rational attackers	0.433	0.321
$\boldsymbol{y}_2$ optimized for bounded rational attackers	0.431	0.388

Table 2: Defender's values in 6×6 gridworld with only decoy allocation.

decoy resources, the attacker reaches the target set with 98.97% probability, and the defender's value is 7.61×10^{-8} at the initial state. If the defender can allocate resources to the decoys, PGA method yields $\boldsymbol{y}_2((0,2)) = 1.141$ and $\boldsymbol{y}_2((5,3)) = 1.0$. The attacker's probability of reaching the target set is 3.99% and the defender's expected value is 0.699. If the defender is allowed to modify the same set of state-action rewards as she is in the previous example, PGA method yields $\boldsymbol{x}_2((4,0), \text{'N'}) = -1, \boldsymbol{x}_2((4,1), \text{'N'}) = -0.85, \boldsymbol{x}_2((4,2), \text{'N'}) = -0.081, \boldsymbol{x}_2((4,4), \text{'N'}) = \boldsymbol{x}_2((4,4), \text{'W'}) = \boldsymbol{x}_2((4,4), \text{'S'}) = -1, \boldsymbol{x}_2((4,4), \text{'E'}) = 0$, $\boldsymbol{y}_2((0,2)) = 0.985$ and $\boldsymbol{y}_2((5,3)) = 1.068$. Under this configuration, the attacker's probability of reaching the target set is 0.3% (93% reduction compared to only allocating decoy resources) and the defender's expected value is 0.730 (4.4% increase compared to only allocate decoy resources). Clearly, the choice of decoy states D influences the attacker's probability of reaching the target set and the defender's expected value: the second set $D' = \{(0,2),(5,3)\}$ appears to outperform the first set $D = \{(1,4),(4,5)\}$. The defender's value is 0.73 given decoy set D', compared to 0.39 given decoy set D.

	No decoy	Decoy only	Decoy and action reward
Attacker's value	0.99	0.099	0.086
Defender's value	3.56×10^{-6}	0.388	0.394

Table 3: Experiment result in 6×6 gridworld given $D = \{(1,4),(4,5)\}$.

	No decoy	Decoy only	Decoy and action reward
Attacker's value	0.99	0.04	0.003
Defender's value	7.61×10^{-8}	0.699	0.730

Table 4: Experiment result in 6×6 gridworld given $D = \{(0,2),(5,3)\}$.

4.1 Scalability

We increase the gridworld size to 10×10 as shown in Figure 3. The sensors, decoy set, and target set are represented using the same notations as the 6×6 gridworld. The results obtained from bi-level LP and PGA are shown in Table 5.

Types of Attackers / Defense Strategies	Rational	Bounded Rational
$\boldsymbol{y}_1$ optimized for rational attackers	0.476	0.469
$\boldsymbol{y}_2$ optimized for bounded rational attackers	0.476	0.472

Table 5: Defender's values in 10×10 gridworld.

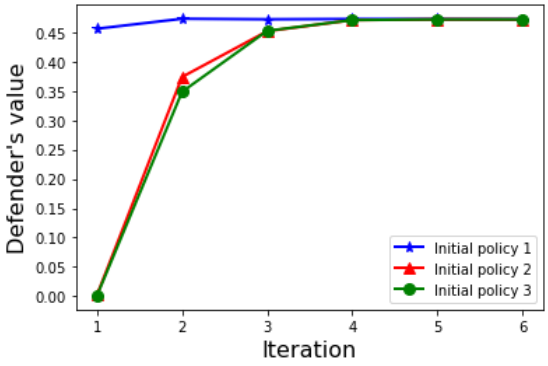

Fig. 4: The convergence of PGA for computing an optimal defense strategy in 10×10 gridworld given different initializations.

We also test the convergence of the PGA method using different initial policies as shown in Figure 4. From Figure 4, we observe that different initial policies result in a similar converged value for the objective function. However, the rate of convergence depends on the initialization of the PGA. The PGA method solved the 10×10 gridworld using 2112.25 seconds and the 6×6 example using 537.58 seconds. The bi-level LP solution running time increases from 0.17 seconds to 0.89 seconds when we increase the gridworld size from 6×6 to 10×10. The running time shows both methods can be extended to moderate problem sizes.

5 Conclusion and Future Work

We present a mathematical framework and algorithms for decoy allocation and reward modification in a proactive defense system against rational and bounded rational attackers. The formulation and solutions can be extended to a broad set of adversarial interactions in which proactive defense with deception can be deployed. In the future, it would be interesting to consider more complex attack and defense objectives and investigate the decoy allocation given the uncertainty in the attacker's goal or capability. Apart from "revealing the fake" studied herein, another direction is to explore how to "conceal the truth" by manipulating the attacker's perceptual reward of compromising true targets.

References

1. Altman, E.: Constrained Markov Decision Processes: Stochastic Modeling. Routledge, Boca Raton, 1 edn. (Dec 2021)
2. Anwar, A.H., Kamhoua, C., Leslie, N.: Honeypot allocation over attack graphs in cyber deception games. In: 2020 International Conference on Computing, Networking and Communications (ICNC). pp. 502–506 (2020)
3. Chen, H.C., Friedman, J.W., Thisse, J.F.: Boundedly Rational Nash Equilibrium: A Probabilistic Choice Approach. Games and Economic Behavior **18**(1), 32–54 (Jan 1997)
4. Dempe, S., Zemkoho, A.: Bilevel optimization. Springer (2020)
5. Durkota, K., Lisỳ, V., Bošanskỳ, B., Kiekintveld, C.: Optimal network security hardening using attack graph games. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)
6. Hansen, P., Jaumard, B., Savard, G.: New branch-and-bound rules for linear bilevel programming. SIAM Journal on scientific and Statistical Computing **13**(5), 1194–1217 (1992)
7. Hong, J., Kim, D.S.: HARMs: Hierarchical Attack Representation Models for Network Security Analysis. In: Australian Information Security Management Conference. p. 9. SRI Security Research Institute, Edith Cowan University, Perth, Western Australia (Dec 2012)
8. Hong, J.B., Kim, D.S.: Assessing the Effectiveness of Moving Target Defenses Using Security Models. IEEE Transactions on Dependable and Secure Computing **13**(2), 163–177 (Mar 2016)
9. Jha, S., Sheyner, O., Wing, J.: Two formal analyses of attack graphs. In: Proceedings 15th IEEE Computer Security Foundations Workshop. CSFW-15. pp. 49–63 (Jun 2002)
10. Lallie, H.S., Debattista, K., Bal, J.: A review of attack graph and attack tree visual syntax in cyber security. Computer Science Review **35**, 100219 (Feb 2020)
11. Madani, O.: On policy iteration as a newton's method and polynomial policy iteration algorithms. In: Eighteenth National Conference on Artificial Intelligence. p. 273–278. American Association for Artificial Intelligence, USA (2002)
12. Mattsson, L.G., Weibull, J.W.: Probabilistic choice and procedurally bounded rationality. Games and Economic Behavior **41**(1), 61–78 (Oct 2002)
13. Milani, S., Shen, W., Chan, K.S., Venkatesan, S., Leslie, N.O., Kamhoua, C., Fang, F.: Harnessing the Power of Deception in Attack Graph-Based Security Games. In: Zhu, Q., Baras, J.S., Poovendran, R., Chen, J. (eds.) Decision and Game Theory for Security. pp. 147–167. Lecture Notes in Computer Science, Springer International Publishing, Cham (2020)
14. Milani, S., Shen, W., Chan, K.S., Venkatesan, S., Leslie, N.O., Kamhoua, C., Fang, F.: Harnessing the power of deception in attack graph-based security games. In: International Conference on Decision and Game Theory for Security. pp. 147–167. Springer (2020)
15. Nachum, O., Norouzi, M., Xu, K., Schuurmans, D.: Bridging the Gap Between Value and Policy Based Reinforcement Learning. In: Advances in Neural Information Processing Systems. vol. 30. Curran Associates, Inc. (2017)
16. Nguyen, T.H., Wright, M., Wellman, M.P., Singh, S.: Multistage Attack Graph Security Games: Heuristic Strategies, with Empirical Game-Theoretic Analysis. Security and Communication Networks **2018**, 1–28 (Dec 2018)
17. Noel, S., Jajodia, S.: Optimal ids sensor placement and alert prioritization using attack graphs. Journal of Network and Systems Management **16**(3), 259–275 (2008)
18. Noel, S., Jajodia, S., Wang, L., Singhal, A.: Measuring security risk of networks using attack graphs. International Journal of Next-Generation Computing **1**(1), 135–147 (2010)
19. Puterman, M.L.: Markov decision processes: discrete stochastic dynamic programming. John Wiley & Sons (2014)

20. Sheyner, O., Haines, J., Jha, S., Lippmann, R., Wing, J.M.: Automated generation and analysis of attack graphs. In: Proceedings 2002 IEEE Symposium on Security and Privacy. pp. 273–284. IEEE (2002)
21. Singhal, A., Ou, X.: Security risk analysis of enterprise networks using probabilistic attack graphs. In: Network Security Metrics, pp. 53–73. Springer (2017)
22. SINHA, A., FANG, F., AN, B., KIEKINTVELD, C., TAMBE, M.: Stackelberg security games: Looking beyond a decade of success. Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18),Stockholm, Sweden, July 13-19 pp. 5494–5501 (Jul 2018)
23. Thakoor, O., Tambe, M., Vayanos, P., Xu, H., Kiekintveld, C., Fang, F.: Cyber Camouflage Games for Strategic Deception. In: Alpcan, T., Vorobeychik, Y., Baras, J.S., Dán, G. (eds.) Decision and Game Theory for Security. pp. 525–541. Lecture Notes in Computer Science, Springer International Publishing, Cham (2019)
24. Ziebart, B.D., Maas, A.L., Bagnell, J.A., Dey, A.K., et al.: Maximum entropy inverse reinforcement learning. In: Aaai. vol. 8, pp. 1433–1438. Chicago, IL, USA (2008)