Tackling Sequential Attacks in Security Games

Thanh H. Nguyen[1], Amulya Yadav[2], Branislav Bosansky[3], Yu Liang[2]

[1] University of Oregon, Eugene, USA
thanhhng@cs.uoregon.edu
[2] Pennsylvania State University, State College, USA
[amulya,luy70]@psu.edu
[3] Czech Technical University in Prague, The Czech Republic
branislav.bosansky@agents.fel.cvut.cz

Abstract. Many real-world security problems exhibit the challenge of sequential attacks (i.e., the attacker carries out multiple attacks in a sequential manner) on important targets. Security agencies have to dynamically allocate limited security resources to the targets in response to these attacks, upon receiving real-time observations regarding them. This paper focuses on tackling sequential attacks using Stackelberg security games (SSGs), a well-known class of leader-follower games, which have been applied for solving many real-world security problems. Previous work on SSGs mainly considers a myopic attacker who attacks one or multiple targets simultaneously against each defense strategy. This paper introduces a new sequential-attack game model (built upon the Stackelberg game model), which incorporates real-time observations, the behavior of sequential attacks, and strategic plans of non-myopic players. Based on the new game model, we propose practical game-theoretic algorithms for computing an equilibrium in different game settings. Our new algorithms exploit intrinsic properties of the equilibrium to derive compact representations of both game state history and strategy spaces of players (which are exponential in number in the original representations). Finally, our computational experiments quantify benefits and losses to the attacker and defender in the presence of sequential attacks.

1 Introduction

In many real-world security domains, security agencies often have to protect important targets such as critical infrastructure from *sequential attacks* carried out by human attackers, given information about these attacks is revealed over time. In fact, an attacker can exploit sequential nature of attacks by setting-up first a decoy attack to attract the attention, following by a more severe attack. Such sophisticated attacks could mislead security agencies to allocate a majority of security resources to handle attacks that have happened, leaving other important targets less protected and thus vulnerable to subsequent attacks. This raises an important question of how to *effectively assign limited security resources* among targets in response to sequential attacks, considering *real-time observations* regarding these attacks.

We propose to use a Stackelberg security game model (SSG) to represent these sequential-attack security scenarios. SSGs have been widely used to model the strategic interaction between the defender and attacker in security domains [1, 3, 10, 13, 15]. In standard SSGs, the attacker is assumed to be a *myopic* player who only attacks once by choosing a single target (or *simultaneously* choosing a subset of targets [8]) against a strategy of the defender. In the sequential attack scenario, the attacker, however, is a *non-myopic* player who considers all future possibilities when deciding on each attack. The attacker can update its belief about the security level at each target after each attack and adapt its next attacks accordingly by leveraging the attacker's prior knowledge of the defender's strategy and partial real-time observations of the resources of the defender.

Our work focuses on addressing the challenge of sequential attacks in security games, with the following main contributions. First, we introduce a new security game model to represent the security problems with sequential attacks. This new model incorporates real-time observations, the behavior of sequential attacks, and strategic plans of players. In the model, the non-myopic attacker carries out multiple attacks in a sequential manner. The attacker strategically adapts its actions based on real-time (partial) observations of defense activities. The defender, on the other hand, has a full observation of previous game states (i.e., which targets were attacked and/or protected) and determines an effective strategic movements of security resources among targets accordingly.

Second, we propose new practical game-theoretic algorithms for computing an equilibrium in two game settings with two rounds of attacks, sorted by the defender's capability of moving security resources after each attack. (i) In the *no-resource-movement* setting, the defender does not move resources when the attacker performs its attacks. This scenario reflects the *worst-case* situation in which the defender is unable to respond as quickly as attacks happen. (ii) In the *resource-movement* setting, the defender can quickly move resources among all targets after each attack. The main computational challenge of finding an equilibrium in these game settings comes from an exponential number of state histories and strategies of players used in the optimization formulation. Our algorithms address this challenge by exploiting intrinsic properties of the equilibrium to derive compact representations of both state histories and strategies.

Finally, we conduct extensive experiments in various game settings to evaluate our proposed algorithm to handle sequential attacks. Our results show that the defender and attacker receive significant loss and benefit respectively if the defender does not address sequential attacks. By taking into account sequential attacks, such loss and benefit is reduced drastically.

2 Background

Stackelberg security games (SSGs) refer to a class of leader-follower games. In standard SSGs, there is a set of important targets $\mathbf{N} = \{1, 2, \ldots, N\}$. A defender has to allocate a limited number of security resources $K < N$ to protect these targets. A pure strategy of the defender is an allocation of these resources over the targets, denoted by $\mathbf{s}$, where each resource protects exactly one target. We denote by $\mathbf{S}$ the set of all these pure strategies. In this work, we consider no scheduling constraint on the defender's resource allocations and all resources are homogeneous and indistinguishable to the players. A mixed strategy of the defender is a randomization over all pure strategies, denoted by $\mathbf{x} = \{x(\mathbf{s})\}$ where $x(\mathbf{s})$ is the probability the defender plays $\mathbf{s}$. We denote by $\mathbf{X} = \{\mathbf{x} : \sum_{\mathbf{s}} x(\mathbf{s}) = 1, 0 \leq x(\mathbf{s}) \leq 1\}$ the set of all mixed strategy of the defender. On the other hand, there is an attacker who is aware of the defender's mixed strategy and aim at attacking one or multiple targets *simultaneously* in response.

When the attacker attacks a target i that is protected by the defender, the attacker receives a penalty P_i^a while the defender obtains a reward of R_i^d. Conversely, the attacker gets a reward of $R_i^a > P_i^a$ and the defender receives a penalty of $P_i^d < R_i^d$. We denote by $\mathbf{S}(i) = \{\mathbf{s} \in \mathbf{S} : i \in \mathbf{s}\}$ which consists of all pure defense strategies that cover target i. The defender and attacker's expected utility at i is computed respectively as follows:

$$U^d(\mathbf{x}, i) = \big[\sum\nolimits_{\mathbf{s}\in\mathbf{S}(i)} x(\mathbf{s})\big](R_i^d - P_i^d) + P_i^d$$
$$U^a(\mathbf{x}, i) = \big[\sum\nolimits_{\mathbf{s}\in\mathbf{S}(i)} x(\mathbf{s})\big](P_i^a - R_i^a) + R_i^a$$

In this work, we call standard `SSG`s as *simultaneous-attack* `SSG`s (i.e., `siSSG`s) to distinguish from our games with sequential attacks. Denote by $L < T$ the number of targets the attacker can attack. Then a simultaneous-attack strategy of the attacker, denoted by $\mathbf{a}_{si}$, is a subset of L targets. Given a mixed strategy of the defender $\mathbf{x}$, if the attacker plays $\mathbf{a}_{si}$, then the attacker and defender's expected utility for playing $(\mathbf{x}, \mathbf{a}_{si})$ is computed as follows:

$$U^a(\mathbf{x}, \mathbf{a}_{si}) = \sum\nolimits_{i\in\mathbf{a}_{si}} U^a(\mathbf{x}, i)$$
$$U^d(\mathbf{x}, \mathbf{a}_{si}) = \sum\nolimits_{i\in\mathbf{a}_{si}} U^d(\mathbf{x}, i)$$

A pair of strategies $(\mathbf{x}_{si}^*, \mathbf{a}_{si}^*)$ of players form a *simultaneous-attack* Strong Stackelberg Equilibrium (`siSSE`) if and only if:

$$\mathbf{x}_{si}^* = \text{argmax}_{\mathbf{x}}\ U^d(\mathbf{x}, \mathbf{a}_{si}^*(\mathbf{x}))$$
$$\mathbf{a}_{si}^*(\mathbf{x}) = \text{argmax}_{\mathbf{a}_{si}}\ U^a(\mathbf{x}, \mathbf{a}_{si})$$

Our work uses `siSSE` as a baseline to compare with our algorithms for tackling sequential attacks. The comparison results are presented in Section 7.

3 Related Work

Security games [7, 14] are a well-known class of resource allocation games where the defender allocates scarce resources to protect selected targets. There are many variants of security games, however, none of them can be used to solve the sequential security game proposed in this paper. The main distinguishing characteristic is a combination of (1) attacker's ability to attack multiple targets and (2) the ability of the attacker (and the defender) to execute their plans sequentially while observing and reacting to the strategy of the opponent.

There are several works that consider scenarios where the attacker can attack multiple targets (defined in this paper as simultaneous-attack SSGs). In the first work, Korzhyk et al. [8] show that computing an SSE in such simultaneous-attack setting is NP-hard, however, their main goal is to design a polynomial-time for finding a Nash equilibrium. The authors, however, do not study the sequential case. Among the follow-up works, the goal of the research is typically allowing more complex scenarios (e.g., allowing dependency among the targets that is not necessarily additive [18]). In theory, one could use such generalized security games where the attacker can attack multiple targets even for modeling the sequential scenario, however, at the cost of an exponential increase in the number of strategies compared to our approach (this is similar to using a normal-form representation in order to solve a sequential game).

On the other hand, in several variants of security games, the players are executing sequential actions. However, the sequential actions are often performed by the defender and not by the attacker. This is the case for applications of green security games [4, 12] or applications against urban crime [19, 5]. In these models, the defender moves sequentially, however, the defender is not able to observe and condition the chosen actions based on the actions of the attacker. This subclass of sequential games has been later generalized for other than security games and domain independent algorithms were provided for the zero-sum case [2, 11]. This is in contrast with our game model where the defender is aware that certain target was attacked in the first step and depending on which target was attacked, the defender can choose different strategy for the next time step.

Finally, security games with sequential attacks can be modeled as general extensive-form games. While computing an SSE in an extensive-form game of this class (with imperfect information) is NP-hard [9], there are several domain-independent algorithms for computing an SSE in extensive-form games that can be, in theory, used for solving security games with sequential attacks [2, 16, 17, 6]. However, extensive-form games do not allow compact representations of strategies and, moreover, all existing algorithms compute solution for strategies with perfect memory. Hence even in a small game with five resources and 10 targets, the defender has 252 possibilities and the game with only two possible attack steps has more than $6 \cdot 10^5$ states. This size corresponds to maximal sizes of games solvable by existing state-of-the-art algorithms for computing a Stackelberg equilibrium in extensive-form games. Contrary, since our novel algorithm is specifically tailored for security games, we are able to achieve much better scalability and we are able to find solutions for more than 20 targets.

4 Sequential-Attack Game Model

Our *sequential-attack* SSG (i.e., seSSG) model is built upon the standard SSG model. Initially, the defender randomly allocates security resources to targets according to a mixed strategy (as defined in `siSSG`s). At the execution time, the defender employs a pure strategy which is sampled from that strategy. The attacker is aware of the mixed strategy of the defender but does not know which targets the defender is protecting at the execution time. In our game, we assume that when the attacker attacks a target i, it can discover if the defender is covering that target or not. Nevertheless, the attacker is still unaware of the current protection status at other targets. By attacking targets sequentially, the attacker is able to explore which targets are being protected by the defender. Based on observations from previous attacks, the attacker can update its belief about the defender's strategy and then decide on targets to attack next that would benefit the attacker the most. In this work, we study the attack scenario in which the attacker can carry out $L > 1$ rounds of attacks and attack one target at each round. The defender has to move security resources among targets in response to such sequential attacks. We assume that when a target is attacked, the damage caused by the attack (if any) to the target is already done. Thus, this target will not be considered in future attack rounds. In addition, if there is a security resource at the attacked target, the resource has to resolve that attack and thus the defender can no longer use that resource for future defense.

4.1 Players' strategies

State and observation history. At each attack round $l \in \{1, 2, ..., L\}$, we denote by $\mathbf{o}_l^d = \{\mathbf{s}_1, i_1, \mathbf{s}_2, i_2, ..., \mathbf{s}_{l-1}, i_{l-1}\}$ the state history of the game. In particular, $\mathbf{s}_{l'}$ is the deployment of security resources and $i_{l'}$ is the attacked target at round $l' < l$. The defender knows $\mathbf{o}_l^d$ while the attacker only has partial observations of the game states. The attacker's observation history is denoted by $\mathbf{o}_l^a = \{(i_1, c(i_1)), (i_2, c(i_2)), ..., (i_{l-1}, c(i_{l-1}))\}$ where $i_{l'}$ is the target attacked at round l' and $c(i_{l'}) \in \{0, 1\}$ represents if the defender is protecting $i_{l'}$ (i.e., $c(i_{l'}) = 1$) or not (i.e., $c(i_{l'}) = 0$). At round 1 specifically, $\mathbf{o}_1^d \equiv \emptyset$ and $\mathbf{o}_1^a \equiv \emptyset$.

Example: As an example, consider a security game with three targets, two defender resources, and two attack rounds. A possible state history at round $l = 2$ is $\mathbf{o}_2^d = \{(1, 2), 2\}$ in which the defender protects targets $\mathbf{s}_1 = (1, 2)$ and the attacker attacks target $i_1 = 2$ at round 1. The corresponding observation history of the attacker is $\mathbf{o}_2^a = \{(2, c(2) = 1)\}$ in which the attacker attacks target $i_1 = 2$ and the defender protects that target (i.e., $c(i_1) = 1$) at round 1.

Defender strategy. At each round l, given a state history $\mathbf{o}_l^d$, the defender (re-)distributes *active* security resources (i.e., resources at targets which have not been attacked) to the targets according to some constraints on resource movements. In the previous example of the 3-target games, in the state history $\mathbf{o}_2^d = \{(1, 2), 2\}$, the defender was protecting target $i_1 = 2$ when the attacker attacks that target at round 1. Therefore, at round $l = 2$, the only remaining active security resource of the defender is located at target 1. Suppose that the defender is able to move that security resource to target 3 at round 2, the defender now has to choose either keep that resource at target 1 or move the resource to target 3. We denote by $\mathbf{S}_l(\mathbf{o}_l^d)$ the set of all possible feasible deployments of security resources at round l given the state history $\mathbf{o}_l^d$. A behavior strategy of the defender at $\mathbf{o}_l^d$, denoted by $\mathbf{x}_l = \{x_l(\mathbf{s}_l \mid \mathbf{o}_l^d)\}$, is a probability distribution over $\mathbf{S}_l(\mathbf{o}_l^d)$ in which $x_l(\mathbf{s}_l \mid \mathbf{o}_l^d)$ is the probability the defender plays the deployment $\mathbf{s}_l$ at round l given the state history $\mathbf{o}_l^d$.

For example, in the 3-target game, given $\mathbf{o}_2^d = \{(1,2), 2\}$, there are two feasible deployments at round $l = 2$: $\mathbf{s}_2 = (1)$ and $\mathbf{s}_2 = (3)$. An example of a behavior strategy of the defender is to protect target 1 and 3 with a probability of 0.6 and 0.4, respectively. At round l, the defender executes a deployment $\mathbf{s}_l$ which is randomly drawn from the strategy $\mathbf{x}_l$. At round 1 specifically, $\mathbf{x}_1 \in \mathbf{X}$ is equivalent to a mixed strategy of the defender in the corresponding `siSSG`.

Attacker strategy. (Stackelberg assumption) We assume the attacker is aware of the defender's behavior strategies, i.e., the probability of each resource deployment of the defender given a state history $\mathbf{o}_l^d$, but not the actual deployments $\mathbf{s}_l$. The attacker decides to attack a target i_l based on its observation history $\mathbf{o}_l^a$.

Bayesian update. At each round $l \in \{1, 2, ..., L\}$, given an observation history $\mathbf{o}_l^a$, the attacker can update its belief regarding the defender's strategy using Bayesian update, formulated as follows:

$$\beta(\mathbf{s}_l \mid \mathbf{o}_l^a) = \sum\nolimits_{\mathbf{o}_l^d} \beta(\mathbf{o}_l^d \mid \mathbf{o}_l^a) x_l(\mathbf{s}_l \mid \mathbf{o}_l^d)$$

$$\beta(\mathbf{o}_l^d \mid \mathbf{o}_l^a) = 0 \text{ if } \mathbf{o}_l^d \text{ and } \mathbf{o}_l^a \text{ are not consistent}$$

$$\beta(\mathbf{o}_l^d \mid \mathbf{o}_l^a) \propto P(\mathbf{o}_l^d) = P(\mathbf{o}_{l-1}^d, \mathbf{s}_{l-1}, i_{l-1}) = x_{l-1}(\mathbf{s}_{l-1} \mid \mathbf{o}_{l-1}^d) P(\mathbf{o}_{l-1}^d), \text{ otherwise}$$

where $\beta(\mathbf{s}_l \mid \mathbf{o}_l^a)$ and $\beta(\mathbf{o}_l^d \mid \mathbf{o}_l^a)$ are the updated belief of the attacker. In particular, $\beta(\mathbf{s}_l \mid \mathbf{o}_l^a)$ is the probability the defender plays $\mathbf{s}_l$ at round l and $\beta(\mathbf{o}_l^d \mid \mathbf{o}_l^a)$ is the probability the state history is $\mathbf{o}_l^d$ given $\mathbf{o}_l^a$. Finally, $\mathbf{o}_l^d$ and $\mathbf{o}_l^a$ are consistent if they share the same attack sequence $(i_1, i_2, \ldots, i_{l-1})$ and $i_{l-1} \in \mathbf{s}_{l-1}$ if $c(i_{l-1}) = 1$ and $i_{l-1} \notin \mathbf{s}_{l-1}$, otherwise. Based on the belief update $\beta(\mathbf{s}_l \mid \mathbf{o}_l^a)$, the attacker will choose next target to attack accordingly.

4.2 Players' utility

Suppose that the defender plays $\mathbf{x}_{se} = \{x_l(\mathbf{s}_l \mid \mathbf{o}_l^d)\}$ (which consists of all behavior strategies of the defender at all of state histories) and the attacker plays $\mathbf{a}_{se} = \{i_l(\mathbf{o}_l^a)\}$ (which consist of all choices of targets to attack at all of observation histories of the attacker), then players' utility at each round can be computed using backward induction as follows:[4]

At round L, given an observation history $\mathbf{o}_L^a$, the attacker's total expected utility for attacking a target $i_L(\mathbf{o}_L^a)$ (shorten by i_L) is computed as follows:

$$U^a(i_L(\mathbf{o}_L^a)) = \left[\sum\nolimits_{\mathbf{s}_L : i_L \in \mathbf{s}_L} \beta(\mathbf{s}_L \,|\, \mathbf{o}_L^a)\right] P_{i_L}^a + \left[\sum\nolimits_{\mathbf{s}_L : i_L \notin \mathbf{s}_L} \beta(\mathbf{s}_L \,|\, \mathbf{o}_L^a)\right] R_{i_L}^a$$

[4] Sometimes we omit $\mathbf{o}_l^a$ and $\mathbf{o}_l^d$ when the context is clear.

On the other hand, given a state history $\mathbf{o}^d_L$, the defender's total expected utility when the attacker attacks a target $i_L(\mathbf{o}^d_L)$ is computed as follows:

$$U^d(i_L(\mathbf{o}^d_L)) = \big[\sum\nolimits_{\mathbf{s}_L : i_L \in \mathbf{s}_L} x_L(\mathbf{s}_L \,|\, \mathbf{o}^d_L)\big] R^d_{i_L} + \big[\sum\nolimits_{\mathbf{s}_L : i_L \notin \mathbf{s}_L} x_L(\mathbf{s}_L \,|\, \mathbf{o}^d_L)\big] P^d_{i_L}$$

where $i_L(\mathbf{o}^d_L) \equiv i_L(\mathbf{o}^a_L)$ with the attacker's observation history $\mathbf{o}^a_L$ is consistent with the state history $\mathbf{o}^d_L$.

At round $l < L$, given an observation history $\mathbf{o}^a_l$, attacker's total expected utility for attacking a target $i_l(\mathbf{o}^a_l)$ (shorten by i_l) is computed as follows:

$$\begin{aligned} U^a(i_l(\mathbf{o}^a_l)) = &\big[\sum\nolimits_{\mathbf{s}_l : i_l \in \mathbf{s}_l} \beta(\mathbf{s}_l \,|\, \mathbf{o}^a_l)\big] \big[P^a_{i_l} + U^a(i_{l+1}(\mathbf{o}^a_l, (i_l, c(i_l) = 1)))\big] \\ &+ \big[\sum\nolimits_{\mathbf{s}_l : i_l \notin \mathbf{s}_l} \beta(\mathbf{s}_l \,|\, \mathbf{o}^a_l)\big] \big[R^a_{i_l} + U^a(i_{l+1}(\mathbf{o}^a_l, (i_l, c(i_l) = 0)))\big] \end{aligned}$$

which comprises of (i) the immediate expected utility for current round and (ii) future expected utility as a result of the current attack. On the other hand, given a state history $\mathbf{o}^d_l$ with a positive probability, the defender's total expected utility when the attacker attacks a target $i_l(\mathbf{o}^d_l)$ is computed as follows:

$$\begin{aligned} U^d(i_l(\mathbf{o}^d_l)) = &\big[\sum\nolimits_{\mathbf{s}_l : i_l \in \mathbf{s}_l} x_l(\mathbf{s}_l \,|\, \mathbf{o}^d_l)\big] R^d_{i_l} + \big[\sum\nolimits_{\mathbf{s}_l : i_l \notin \mathbf{s}_l} x_l(\mathbf{s}_l \,|\, \mathbf{o}^d_l)\big] P^d_{i_l} \\ &+ \sum\nolimits_{\mathbf{s}_l} x_l(\mathbf{s}_l \mid \mathbf{o}^d_l) U^d(i_{l+1}(\mathbf{o}^d_l, \mathbf{s}_l, i_l(\mathbf{o}^d_l))) \end{aligned}$$

where $i_l(\mathbf{o}^d_l) \equiv i_l(\mathbf{o}^a_l)$ with the observation history of the attacker $\mathbf{o}^a_l$ is consistent with the state history $\mathbf{o}^d_l$.

Finally, players' total expected utility for playing $(\mathbf{x}_{se}, \mathbf{a}_{se})$ is determined as:[5]

$$U^d(\mathbf{x}_{se}, \mathbf{a}_{se}) = U^d(i_1)$$
$$U^a(\mathbf{x}_{se}, \mathbf{a}_{se}) = U^a(i_1)$$

4.3 Sequential-attack SSE

A pair of strategies of players $(\mathbf{x}^*_{se} = \{x^*_l(\mathbf{s}_l \mid o^d_l)\}, \mathbf{a}^*_{se}(\mathbf{x}^*) = \{i^*_l(\mathbf{o}^a_l)\})$ forms a sequential-attack SSE (`seSSE`) if and only if:

$$\mathbf{x}^*_{se} = \operatorname*{argmax}_{\mathbf{x}_{se}} U^d(\mathbf{x}_{se}, \mathbf{a}^*_{se}(\mathbf{x}_{se}))$$
$$\mathbf{a}^*_{se}(\mathbf{x}_{se}) = \operatorname*{argmax}_{\mathbf{a}_{se}} U^a(\mathbf{x}_{se}, \mathbf{a}_{se}).$$

In this paper, we study `seSSG`s with the focus on computing an `seSSE` of the game in two game settings, sorted by the defender's capability of moving security resources after each attack. (i) In the *resource-movement* setting, the defender can move security resource from a target to *any* other target after each attack. This setting captures the situation in which the defender has a full capability

[5] For the sake of presentation, we will omit $\emptyset$ (i.e., $i_1 = i_1(\emptyset)$) from all contexts.

of re-allocating resources in response to every attack. (i) In the *no-resource-movement* setting, the defender does not move resources when attacks happens. This setting reflects the *worst* security scenario in which the defender is unable to react (by moving resources) as quickly as attacks happening.

Analyzing and finding an `seSSE` in general is computationally expensive which involves exponentially many strategies of both the attacker and the defender, as well as exponentially many state histories. Therefore, in this paper, we focus on developing efficient game-theoretic algorithms to compute an `seSSE` in these game settings in which the attacker attacks two targets sequentially, i.e., $L = 2$. In fact, as we shown later, even in this 2-round game scenario, the search space is still exponential. We provide efficient algorithms to compute an `seSSE`, by (i) exploiting underlying characteristics of `seSSG`s to compactly represent spaces of state histories and strategies of players; and (ii) applying optimization techniques such as cutting plane to scale up the computation of an `seSSE`.

5 The Resource-Movement Setting

In the resource movement game setting with two (sequential) attacks, the defender has to determine: (i) how to allocate resources before any attacks happens; and (ii) how to move resources in response to the first attack.

5.1 Compact representation

Solving the 2-round `seSSE` is computationally expensive since it involves an exponential number of resource allocations and movements of the defender. In fact, the exponentially many resource allocations at first round also leads to an exponential number of state histories at second round. In the following, we introduce an equivalent compact representation of the players' strategies and state histories, leveraging the *resource-movement* property. We first present a characteristic of the `seSSE` in Theorem 1.

Theorem 1. *In the 2-round* `seSSG`*s with resource movements, the defender's strategy at the second round in the* `seSSE` *can be compactly represented such that the compact representation only depends on* which target is attacked at the first round and whether the defender is protecting that target or not.

Proof. Consider the `seSSE` of the game $(\mathbf{x}^*_{se}, \mathbf{a}^*_{se})$. According to the definition of the `seSSE` in Section 4.3, given the players' strategy at the first round $(\{x^*_1(\mathbf{s}_1)\}, i^*_1)$ in the `seSSE`, the defender's strategy at the second round is the optimal solution of the following optimization problem:

$$\max_{\mathbf{x}_2} \sum_{\mathbf{s}_1 : i^*_1 \in \mathbf{s}_1} x^*_1(\mathbf{s}_1) \sum_{\mathbf{s}_2} x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i^*_1) V^d(\mathbf{s}_2, i^*_2(i^*_1, c(i^*_1) = 1)) \tag{1}$$

$$+ \max_{\mathbf{x}_2} \sum_{\mathbf{s}_1 : i^*_1 \notin \mathbf{s}_1} x^*_1(\mathbf{s}_1) \sum_{\mathbf{s}_2} x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i^*_1) V^d(\mathbf{s}_2, i^*_2(i^*_1, c(i^*_1) = 0)) \tag{2}$$

which maximizes the defender's expected utility at second round. The term $V^d(\mathbf{s}_2, i_2^*(i_1^*, c(i_1^*) = 1))$ is equal to the defender's reward at target $i_2^*(i_1^*, c(i_1^*) = 1)$ if the deployment of defender resources $\mathbf{s}_2$ covers that target. Otherwise, it is equal to the defender's penalty at the target. The first optimization component in (1) can be equivalently represented as follows:

$$\left[\sum_{\mathbf{s}_1 : i_1^* \in \mathbf{s}_1} x_1^*(\mathbf{s}_1)\right] \max_{\mathbf{x}_2} \sum_{\mathbf{s}_2, \mathbf{s}_1 : i_1^* \in \mathbf{s}_1} \frac{x_1^*(\mathbf{s}_1)}{\sum_{\mathbf{s}_1' : i_1^* \in \mathbf{s}_1'} x_1^*(\mathbf{s}_1')} x_2(\mathbf{s}_2 \,|\, \mathbf{s}_1, i_1^*) V^d(\mathbf{s}_2, i_2^*(i_1^*, c(i_1^*) = 1))$$

Also, the attacker's second attack $i_2^*(i_1^*, c(i_1^*) = 1)$ is an optimal solution of:

$$\left[\sum\nolimits_{\mathbf{s}_1 : i_1^* \in \mathbf{s}_1} x_1^*(\mathbf{s}_1)\right] \max_{i_2 \neq i_1^*} \sum\nolimits_{\mathbf{s}_2, \mathbf{s}_1 : i_1^* \in \mathbf{s}_1} \beta(\mathbf{s}_1 \mid i_1^*, c(i_1^*) = 1) x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i_1^*) V^a(\mathbf{s}_2, i_2)$$

which maximizes the attacker's expected utility at second round. The term $V^a(\mathbf{s}_2, i_2)$ is equal to the attacker's reward at i_2 if the target is not covered by the defender's deployment $\mathbf{s}_2$. Otherwise, it is equal to the attacker's penalty at that target. The attacker's updated belief is computed using Bayesian update:

$$\beta(\mathbf{s}_1 \mid i_1^*, c(i_1^*) = 1) = \frac{x_1^*(\mathbf{s}_1)}{\sum_{\mathbf{s}_1' : i_1^* \in \mathbf{s}_1'} x_1^*(\mathbf{s}_1')}, \text{ if } i_1^* \in \mathbf{s}_1$$

We introduce the following new variables:

$$y^{i_1^*,1}(\mathbf{s}_2) = \sum\nolimits_{\mathbf{s}_1 : i_1^* \in \mathbf{s}_1} \frac{x_1^*(\mathbf{s}_1)}{\sum_{\mathbf{s}_1' : i_1^* \in \mathbf{s}_1'} x_1^*(\mathbf{s}_1')} x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i_1^*), \forall \mathbf{s}_2$$

Since the defender can move resource from one target to *any* other targets without any constraint, any values of $\mathbf{y}^{i_1^*,1} = \{y^{i_1^*,1}(\mathbf{s}_2)\}$ such that:

$$\sum\nolimits_{\mathbf{s}_2} y^{i_1^*,1}(\mathbf{s}_2) = 1, y^{i_1^*,1}(\mathbf{s}_2) \in [0,1], \forall \mathbf{s}_2$$

is equivalent to a strategy of the defender $\{x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i_1^*)\}$ at second round (i.e., we can simply assign $x_2(\mathbf{s}_2 \mid \mathbf{s}_1, i_1^*) = y^{i_1^*,1}(\mathbf{s}_2)$ for all $\mathbf{s}_2$). Therefore, for the rest of this section, we can consider $\mathbf{y}^{i_1^*,1} = \{y^{i_1^*,1}(\mathbf{s}_2)\}$ as a strategy of the defender at second round given the attacker attacks target i_1^* at first round and the defender is covering that target. The first optimization component in (1) is now equivalent to the following optimization problem:

$$\begin{aligned}
&\max\nolimits_{\mathbf{y}^{i_1^*,1}} \sum\nolimits_{\mathbf{s}_2} y^{i_1^*,1}(\mathbf{s}_2) V^d(\mathbf{s}_2, i_2^*(i_1^*, c(i_1^*) = 1)) \\
&\quad \text{s.t. } i_2^*(i_1^*, c(i_1^*) = 1) = \operatorname{argmax}_{i_2 \neq i_1^*} \sum\nolimits_{\mathbf{s}_2} y^{i_1^*,1}(\mathbf{s}_2) V^a(\mathbf{s}_2, i_2) \\
&\quad \sum\nolimits_{\mathbf{s}_2} y^{i_1^*,1}(\mathbf{s}_2) = 1, y^{i_1^*,1}(\mathbf{s}_2) \in [0,1]
\end{aligned}$$

which results in a `siSSE` of a `siSSG` which consists of (i) a target set $\mathbf{N} \setminus \{i_1^*\}$; (ii) $K - 1$ security resources, and (iii) one attack. Similarly, we can also show that the second optimization component in (2) corresponds to a `siSSE` of the similar game but with K security resources. These `siSSE`s only depends on which target is attacked and if the defender protects that target or not at first round. □

Based on Theorem 1, each strategy of the defender can be now compactly represented as having two components: (i) $\{x_1(\mathbf{s}_1)\}$ where $x_1(\mathbf{s}_1)$ is the probability the defender plays $\mathbf{s}_1 \in \mathbf{S}$ before any attacks; and (ii) $(\{y^{i,1}(\mathbf{s}_2)\}; \{y^{i,0}(\mathbf{s}_2)\})$ where $y^{i,1}(\mathbf{s}_2)$ is the probability the defender plays $\mathbf{s}_2 \in \mathbf{S}^{K-1}(-i)$ if the first attack is towards target i and the defender is protecting that target. The set $\mathbf{S}^{K-1}(-i)$ is the set of subsets of $K-1$ targets (there are $K-1$ resources left) excluding target i. In addition $y^{i,0}(\mathbf{s}_2)$ is the probability the defender plays $\mathbf{s}_2 \in \mathbf{S}^{K}(-i)$ if the first attack is towards target i and while the defender is not protecting this target (the defender allocates K resources to targets excluding target i).

Note that the current compact representation of the defender's strategies still involves all possible deployments of the defender's security resources at each round, of which number is exponential. We then provide Proposition 1 (its proof is in the Online appendix A)[6] showing that the defender strategies at each round are equivalent to compact marginal coverage probabilities at every target.

Proposition 1. *In the sequential-attack game with resource movements, the defender's strategies can be compactly represented as follows:*

$$\begin{aligned}
x_j &= \sum\nolimits_{\mathbf{s}_1} x_1(\mathbf{s}_1), \sum\nolimits_j x_j = K, \forall j \\
y_j^{i,1} &= \sum\nolimits_{\mathbf{s}_2} y^{i,1}(\mathbf{s}_2), \sum\nolimits_{j \neq i} y_j^{i,1} = K-1, \forall i, j \neq i \\
y_j^{i,0} &= \sum\nolimits_{\mathbf{s}_2} y^{i,0}(\mathbf{s}_2), \sum\nolimits_{j \neq i} y_j^{i,0} = K, \forall i, j \neq i
\end{aligned}$$

where x_j is the marginal probability the defender protects target j before any attack happens. In addition, $y_j^{i,1}$ and $y_j^{i,0}$ are the marginal probabilities the defender protects target $j \neq i$ after the attacker attacked target i while the defender was protecting and was not protecting that target, respectively.

5.2 Mixed integer linear program (MILP) representation

According to Theorem 1, given the attacked target i_1^* at round 1, the player's equilibrium strategies in round 2 is equivalent to an `siSSE` of a `siSSG`, which consists of (i) a target set $\mathbf{N} \setminus \{i_1^*\}$; (ii) K defender resources (if the defender is not protecting i_1^* in the original `seSSG`) or $K-1$ resources (otherwise); and (iii) one attack. This `siSSE` can be computed in advance. Therefore, we introduce the following MILP to compute the `seSSE` of the 2-round `seSSG` based on these pre-computed `siSSE`s and the compact representation described in Proposition 1:

$$\begin{aligned}
&\max v \\
\text{s.t. } & v \leq x_i(R_i^d - P_i^d) + P_i^d + (1-x_i)U^d_{\texttt{siSSE}}(i,0) + x_i U^d_{\texttt{siSSE}}(i,1) + (1-h_i)M, \forall i \\
& r \geq x_i(P_i^a - R_i^a) + R_i^a + (1-x_i)U^a_{\texttt{siSSE}}(i,0) + x_i U^a_{\texttt{siSSE}}(i,1), \forall i \\
& r \leq x_i(P_i^a - R_i^a) + R_i^a + (1-x_i)U^a_{\texttt{siSSE}}(i,0) + x_i U^a_{\texttt{siSSE}}(i,1) + (1-h_i)M, \forall i \\
& \sum\nolimits_i h_i = 1, h_i \in \{0,1\}
\end{aligned}$$

[6] Online appendix: https://www.dropbox.com/s/hjyjabfg69llyn3/Appendix.pdf?dl=0

where $U^d_{\texttt{siSSE}}(i,0)$ and $U^d_{\texttt{siSSE}}(i,1)$ are the defender's utilities in the `siSSE`s of the resulting `siSSG`s after the attacker attacks i while the defender is not protecting and protecting the target, respectively. Similarly, $U^a_{\texttt{siSSE}}(i,0)$ and $U^a_{\texttt{siSSE}}(i,1)$ are the attacker's equilibrium utilities in the resulting `siSSE`s. In addition, v is the defender's total expected utility which we aim to maximize and r is the attacker's total expected utility. The binary variable h_i represent if the attacker attacks target i ($h_i = 1$) or not ($h_i = 0$) at the first round.

6 The No-Resource-Movement Setting

In this section, we study the problem in which the defender does not move resources when the attacker performs its attacks. This setting reflects the *response-delayed* security scenario in which the defender is unable to react (by moving resources) as quickly as attacks happening. In this scenario, the defender's goal is to optimize his randomization over resource allocations before such sequential attacks happen. Therefore, we use the same notation $\mathbf{x} = \{x(\mathbf{s})\}$ as in the *simultaneous-attack* games to represent the defender's mixed strategy. An attack strategy of the attack is denoted by $\mathbf{a}_{se}$ which is defined the same as in the unconstrained resource movement setting. The `seSSE` is still denoted by $(\mathbf{x}^*_{se}, \mathbf{a}^*_{se})$.

6.1 Equilibrium analysis

We first provide Theorem 2 showing the benefit and loss of the attacker and defender for playing sequentially instead of simultaneously in zero-sum games.

Theorem 2. *In zero-sum games, the attacker obtains a higher utility while the defender gets a lower utility from sequential attacks than from simultaneous attacks. In particular, we have:*

$$U^d(\mathbf{x}^*_{se}, \mathbf{a}^*_{se}(\mathbf{x}^*_{se})) \leq U^d(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si})) \tag{3}$$

$$U^a(\mathbf{x}^*_{se}, \mathbf{a}^*_{se}(\mathbf{x}^*_{se})) \geq U^a(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si})) \tag{4}$$

Proof. (i) First, given any defense strategy $\mathbf{x}$, the attacker always obtains a higher total expected utility for playing sequentially than simultaneously. Indeed, we denote the simultaneous-attack best response $\mathbf{a}^*_{si} = (i^*, j^*)$. The corresponding expected utility of the attacker is $U^a(\mathbf{x}, i^*) + U^a(\mathbf{x}, j^*)$. On the other hand, finding a sequential-attack best response is determined as follows:

$$\max_{\mathbf{a}_{se}} U^a(\mathbf{x}, \mathbf{a}_{se}) \geq U^a(\mathbf{x}, i^*) + U^a(\mathbf{x}, j^*).$$

Essentially, the attacker's simultaneous-attack best response (i^*, j^*) corresponds to a feasible sequential-attack response in which the attacker attacks i^* at first attack and then attacks j^* at second attack regardless of the attack result of the first attack. Therefore, $U^a(\mathbf{x}^*_{se}, \mathbf{a}^*_{se}(\mathbf{x}^*_{se})) \geq U^a(\mathbf{x}^*_{se}, \mathbf{a}^*_{si}(\mathbf{x}^*_{se}))$.

(ii) In addition, based on the definition of the `siSSE`, we have the defender's utility $U^d(\mathbf{x}^*_{se}, \mathbf{a}^*_{si}(\mathbf{x}^*_{se})) \leq U^d(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si}))$. This inequality is equivalent to $U^a(\mathbf{x}^*_{se}, \mathbf{a}^*_{si}(\mathbf{x}^*_{se})) \geq U^a(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si}))$ (according to the zero-sum property).

Based on (i) and (ii), we have $U^a(\mathbf{x}^*_{se}, \mathbf{a}^*_{se}(\mathbf{x}^*_{se})) \geq U^a(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si}))$. Since this game is zero-sum, we obtain: $U^d(\mathbf{x}^*_{se}, \mathbf{a}^*_{se}(\mathbf{x}^*_{se})) \leq U^d(\mathbf{x}^*_{si}, \mathbf{a}^*_{si}(\mathbf{x}^*_{si}))$. □

Computing an `seSSE` is computationally expensive due to an exponential number of pure strategies of the defender. In the following, we present our MILP formulation to exactly compute an `seSSE`. We then provide a scalable algorithm which is based on the compact representation and cutting-plane based method to overcome the computation challenge. We denote by $\mathbf{S}(i)$ and $\mathbf{S}(-i)$ the sets of pure strategies of the defender which cover and do not cover target i, respectively. We denote by $\mathbf{S}(i,j)$ the set of pure strategies which cover both (i,j).

6.2 Equilibrium computation: MILP formulation

We first present Lemma 1, showing the linearity relationship between the players' total expected utility and the defender's mixed strategies. This result serves as a basis to develop an MILP to exactly compute an `seSSE`.

Lemma 1. *Given an attack strategy $\mathbf{a}_{se}$, both the attacker and defender's total expected utility is a linear function of the defender's mixed strategy $\mathbf{x}$.*

Proof. Based on the computation of the players' utility described in Section 4, we can represent the attacker's total expected utility as follows:

$$\begin{aligned} U^a(\mathbf{x}, \mathbf{a}_{se}) = &\big[\textstyle\sum_{\mathbf{s}\in\mathbf{S}(i_1)} x(\mathbf{s})\big](P^a_{i_1} - R^a_{i_1}) + R^a_{i_1} \\ &+ \big[\sum_{\mathbf{s}\in\mathbf{S}(i_1)} x(\mathbf{s})\big]\big[\sum_{\mathbf{s}\in\mathbf{S}(i_1, i_2(i_1,1))} \beta(\mathbf{s} \mid i_1, c(i_1)=1)(P^a_{i_2(i_1,1)} - R^a_{i_2(i_1,1)}) + R^a_{i_2(i_1,1)}\big] \\ &+ \big[\sum_{\mathbf{s}\in\mathbf{S}(-i_1)} x(\mathbf{s})\big]\big[\sum_{\mathbf{s}\in\mathbf{S}(-i_1, i_2(i_1,0))} \beta(\mathbf{s} \mid i_1, c(i_1)=0)(P^a_{i_2(i_1,0)} - R^a_{i_2(i_1,0)}) + R^a_{i_2(i_1,0)}\big] \end{aligned} \tag{5}$$

where the attacker's updated belief is computed using the Bayesian update:

$$\beta(\mathbf{s} \mid i_1, c(i_1)=1) = \frac{x(\mathbf{s})}{\sum_{\mathbf{s}'\in\mathbf{S}(i_1)} x(\mathbf{s}')}, \text{ if } \mathbf{s} \in \mathbf{S}(i_1)$$

The updated belief, $\beta(\mathbf{s} \mid i_1, c(i_1)=0)$, is computed similarly. Therefore, the attacker's total expected utility can be formulated as follows:

$$\begin{aligned} U^a(\mathbf{x}, \mathbf{a}_{se}) = &\big[\textstyle\sum_{\mathbf{s}\in\mathbf{S}(i_1)} x(\mathbf{s})\big](P^a_{i_1} - R^a_{i_1}) + R^a_{i_1} \\ &+ \big[\sum_{\mathbf{s}\in\mathbf{S}(i_1, i_2(i_1,1))} x(\mathbf{s})\big](P^a_{i_2(i_1,1)} - R^a_{i_2(i_1,1)}) + \big[\sum_{\mathbf{s}\in\mathbf{S}(i_1)} x(\mathbf{s})\big] R^a_{i_2(i_1,1)} \\ &+ \big[\sum_{\mathbf{s}\in\mathbf{S}(-i_1, i_2(i_1,0))} x(\mathbf{s})\big](P^a_{i_2(i_1,0)} - R^a_{i_2(i_1,0)}) + \big[\sum_{\mathbf{s}\in\mathbf{S}(-i_1)} x(\mathbf{s})\big] R^a_{i_2(i_1,0)} \end{aligned} \tag{6}$$

which is a linear function of $\mathbf{x}$. Finally, we apply the same computation process to show that the defender's total expected utility is also linear in $\mathbf{x}$. □

In the `seSSE`, our goal is to find an optimal strategy for the defender that maximizes the defender's total expected utility. Based on Lemma 1, this problem can be represented as an MILP, formulated as follows:

$$\max_{\mathbf{x},\mathbf{a}_{se}} v \text{ s.t.} \tag{7}$$

$$v \leq U^d(\mathbf{x},\mathbf{a}_{se}) + (3 - h_i - q_j^1 - q_k^0)M, \forall i \neq j, k \tag{8}$$

$$r \geq U^a(\mathbf{x},\mathbf{a}_{se}), \forall i \neq j, k \tag{9}$$

$$r \leq U^a(\mathbf{x},\mathbf{a}_{se}) + (3 - h_i - q_j^1 - q_k^0)M, \forall i \neq j, k \tag{10}$$

$$\sum\nolimits_i h_i = 1, h_i \in \{0,1\} \tag{11}$$

$$\sum\nolimits_j q_j^0 = 1, \sum\nolimits_j q_j^1 = 1, q_j^0, q_j^1 \in \{0,1\} \tag{12}$$

$$h_i + q_i^0 \leq 1, h_i + q_i^1 \leq 1. \tag{13}$$

where $\mathbf{a}_{se}$ is defined as $i_1 = i$, $i_2(i_1, 1) = j$ and $i_2(i_1, 0) = k$ (i.e., the attacker will attack target j at round 2 if the defender protects target i at round 1 and attack target k, otherwise). We first introduce three binary variables: (i) $h_i = 1$ if the attacker attacks target i at first round and $h_i = 0$, otherwise; (ii) $q_j^0 = 1$ if the attacker attacks target j at second round given the defender is not protecting the attacked target at first round and $q_j^0 = 0$, otherwise; and (iii) $q_j^1 \in \{0,1\}$ given the defender is protecting the attacked target at first round. In the MILP, v and r are the defender and attacker's total expected utility. Constraint (8) determines the defender's utility given a best response of the attacker (when $h_i = 1, q_j^1 = 1, q_k^0 = 1$). Constraints (9–10) ensure that r is the attacker's utility with respect to the attacker's best response (when $h_i = 1, q_j^1 = 1, q_k^0 = 1$). Constraints (11–12) imply the attacker attacks one target at each round. Finally constraint (13) imposes that if the attacker already attacks a target i, it will not attack i again. Finally, M is a large constant which ensures that associated constraints are effective only when $h_i = 1, q_j^1 = 1, q_j^0 = 1$.

While the MILP provide an exact `seSSE`, it has limited scalability due to an exponential number of pure strategies of the defender in $\mathbf{S}$. Therefore, we introduce a new efficient algorithm to overcome this computation issue.

6.3 Equilibrium computation: scalable algorithm

Our new algorithm comprises of two ideas: (i) *compact representation* and (ii) a *cutting-plane* method. For the first idea, we provide a *relaxed* compact representation of the defender's mixed strategies. This compact representation may not be equivalent to a feasible mixed strategy. Yet, the number of *compact* defense strategies is significantly small (i.e., $O(|N|^2)$) compared to the original representation with the number of strategies is $\binom{N}{K}$. For the second idea, we apply the cutting-plane method to gradually refine the compact strategy space. This iterative process stops when the optimal strategy of the defender found in the refined compact space is feasible in the original space, which means an optimal mixed strategy of the defender is found.

Compact representation. We propose the following compact representation:

$$x_{i,j} = \sum\nolimits_{\mathbf{s}\in\mathbf{S}(i,j)} x(\mathbf{s})$$

where $x_{i,j}$ represents the probability the defender protects both target i and j simultaneously. There is the following resource constraint associated with $\{x_{i,j}\}$:

$$\sum\nolimits_i \sum\nolimits_{j\neq i} x_{i,j} = 2 \times \sum\nolimits_{i<j} x_{i,j} = K \times (K-1) \tag{14}$$

For each pair (i, j), the probability $x(\mathbf{s})$ where $\mathbf{s} \in \mathbf{S}(i, j)$ is counted once in computing $x_{i,j}$. For each pure strategy $\mathbf{s}$ of the defender, there are $\frac{K\times(K-1)}{2}$ pairs of targets from $\mathbf{s}$. Therefore, $x(\mathbf{s})$ is counted $\frac{K\times(K-1)}{2}$ times in computing $\sum_{i<j} x_{i,j}$. As a result, we obtain Eq. 14.

Based on this compact representation, we can compute the players' total expected utility shown in Lemma 1 accordingly. In particular,

$$\sum\nolimits_{\mathbf{s}\in\mathbf{S}(i)} x(\mathbf{s}) = \frac{1}{K-1}\sum\nolimits_{j\neq i} x_{i,j} \tag{15}$$

$$\sum\nolimits_{\mathbf{s}\in\mathbf{S}(-i)} x(\mathbf{s}) = 1 - \sum\nolimits_{\mathbf{s}\in\mathbf{S}(i)} x(\mathbf{s}) = 1 - \frac{1}{K-1}\sum\nolimits_{j\neq i} x_{i,j} \tag{16}$$

$$\sum\nolimits_{\mathbf{s}\in\mathbf{S}(-i,j)} x(\mathbf{s}) = \sum\nolimits_{\mathbf{s}\in\mathbf{S}(j)} x(\mathbf{s}) - x_{i,j} = \left(\frac{1}{K-1}\sum\nolimits_{k\neq j} x_{j,k}\right) - x_{i,j} \tag{17}$$

We now can use $\{x_{i,j}\}$ instead of $\{x(\mathbf{s})\}$ to solve (7–13). However, $\{x_{i,j}\}$ satisfying the aforementioned constraints is generally not equivalent to a feasible mixed strategy of the defender in the original strategy space. For example, let's consider a security game with 4 targets $\{1, 2, 3, 4\}$ and 3 defender resources. There are four pure strategies of the defender $\{(1, 2, 3), (1, 2, 4), (1, 3, 4), (2, 3, 4)\}$. A possible compact strategy is $(x_{1,2}; x_{1,3}; x_{1,4}; x_{2,3}; x_{2,4}; x_{3,4}) = (1.0; 1.0; 0.0; 0.0; 1.0; 0.0)$. According to this compact strategy, the defender will protect target 1 with a probability of $\frac{x_{12}+x_{13}+x_{14}}{2} = 1.0$ and target 2 with a probability of $\frac{x_{12}+x_{23}+x_{24}}{2} = 1.0$. This implies that the defender only plays either $(1, 2, 3)$ or $(1, 2, 4)$. Thus,

$$1.0 = x_{13} = x_{123} + x_{134} = x_{123}$$
$$1.0 = x_{24} = x_{124} + x_{234} = x_{124}$$

and as a result, $x_{123} + x_{124} + x_{134} + x_{234} \geq 2.0$ which is infeasible since the probabilities of playing defense pure strategies must sum up to 1.0.

Therefore, we propose to use the cutting plane method (explained below) to gradually refine the solution space of $\{x_{i,j}\}$ in solving the MILP (7–13).

Cutting plane method. At a high level, we solve the MILP (7–13) with a set of additional linear constraints on $\{x_{i,j}\}$. These constraints impose the feasible value domain of $\{x_{i,j}\}$, which are determined iteratively through the cutting plane method. Initially, we solve (7–13) without any constraint on $\{x_{i,j}\}$

and obtain a (possibly infeasible) solution $\{\bar{x}_{i,j}\}$. We then apply the cutting plane method to find the plane (deep cut) which separates the feasible domain of $\{x_{i,j}\}$ and the (infeasible) point $\{\bar{x}_{i,j}\}$. As a result, we obtain a new linear constraint which refines the solution space of $\{x_{i,j}\}$. We solve the MILP again with the additional constraint. This process will continue until we reach an optimal mixed strategy for the defender. More specifically, given an outcome of $\{\bar{x}_{i,j}\}$, we consider the following feasibility problem

$$\min_{\{x(\mathbf{s})\}} \sum_{i,j} d_{ij} \tag{18}$$

$$\text{s.t. } d_{ij} \geq \bar{x}_{i,j} - \sum\nolimits_{\mathbf{s}\in\mathbf{S}(i,j)} x(\mathbf{s}), \forall i \neq j \tag{19}$$

$$d_{ij} \geq \sum\nolimits_{\mathbf{s}\in\mathbf{S}(i,j)} x(\mathbf{s}) - \bar{x}_{i,j}, \forall i \neq j \tag{20}$$

$$\sum\nolimits_{\mathbf{s}\in\mathbf{S}} x(\mathbf{s}) = 1, x(\mathbf{s}) \geq 0, \forall \mathbf{s} \in \mathbf{S} \tag{21}$$

which finds the feasible mixed strategy closest to the compact strategy $\{\bar{x}_{i,j}\}$. The term $\sum_{i,j} d_{i,j}$ represents the 1-norm distance between $\{x(\mathbf{s})\}$ and $\{\bar{x}_{i,j}\}$ which we aim to minimize. Denote by $(d^*, \{x^*(\mathbf{s})\})$ the optimal solution of (18–21), we obtain the following proposition, determining if $\{\bar{x}_{i,j}\}$ is feasible or not.

Proposition 2. *If $d^* = 0$, the compact solution $\{\bar{x}(i,j)\}$ returns an optimal defense mixed strategy. If $d^* > 0$, $\{\bar{x}_{i,j}\}$ is not feasible.*

Let's consider the dual problem of (18–21):

$$\max \sum\nolimits_{i\neq j} \bar{x}_{ij}(y_{ij} - z_{ij}) + \lambda \tag{22}$$

$$\text{s.t.} \sum\nolimits_{i,j\in\mathbf{s}} y_{ij} - \sum\nolimits_{i,j\in\mathbf{s}} z_{ij} + \lambda \leq 0, \forall \mathbf{s} \tag{23}$$

$$y_{ij} + z_{ij} \leq 1, \forall i \neq j \tag{24}$$

$$y_{ij}, z_{ij} \geq 0. \tag{25}$$

We denote by $\{y^*_{ij}, z^*_{ij}, \lambda^*\}$ the dual optimal solution. Proposition 3 provides a cutting plane (i.e., a linear constraint) which refines the compact solution space.

Proposition 3. *Given an infeasible $\{\bar{x}_{ij}\}$, the cutting plane $\sum_{i\neq j} x_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* = 0$ separates $\{\bar{x}_{ij}\}$ from the mixed strategy space of the defender:*

1. $\sum_{i\neq j} \bar{x}_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* > 0$.
2. $\sum_{i\neq j} x_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* \leq 0, \forall$ *feasible* $\{x_{ij}\}$.

Proof. Since $\{\bar{x}_{ij}\}$ is infeasible, then $\sum_{i\neq j} x_{ij}(y_{ij} - z_{ij}) + \lambda > 0$ for all feasible $\{y_{ij}, z_{ij}, \lambda\}$ of the dual program. Therefore, we have $\sum_{i\neq j} x_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* > 0$.

For all feasible $\{x_{ij}\}$, the optimal objective of the dual program with respect to $\{x_{ij}\}$, denoted by $\{y'_{ij}, z'_{ij}, \lambda'\}$, must be equal to zero. Thus, we obtain:
$\sum_{i\neq j} x_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* \leq \sum_{i\neq j} x_{ij}(y'_{ij} - z'_{ij}) + \lambda' = 0.$ □

Based on Proposition 3, the new linear constraint to add to the problem (7–13) is $\sum_{i \neq j} x_{ij}(y^*_{ij} - z^*_{ij}) + \lambda^* \leq 0$, which refine the search space for compact strategies $\{x_{ij}\}$. We now aim at solving (22–25) to find the cutting plane. This dual program involves an exponential number of constraints since there is an exponential number of pure strategies for the defender (Constraint 23). Therefore, we propose to use the incremental constraint generation approach (i.e., column generation). That is, we solve the relaxed dual program with respect to a small subset of mixed defense strategy in $\mathbf{S}$. We then gradually add new constraints until we obtain the optimal solution (i.e., no violated constraint is found). The main part of this approach is to find a maximally violated constraint given the current optimal solution $\{y^*_{ij}, z^*_{ij}, \lambda^*\}$ of the relaxed (22–25). This problem is equivalent to finding a pure strategy $\mathbf{s} \in \mathbf{S}$ such that $\sum_{i,j \in \mathbf{s}} y^*_{ij} - \sum_{i,j \in \mathbf{s}} z^*_{ij} + \lambda^*$ is maximum. Intuitively, we want to find $\mathbf{s}$ such that the constraint (23) is violated the most, which can be represented as the following MILP:

$$\begin{aligned} \max & \sum\nolimits_{i,j} y^*_{ij} h_{ij} - \sum\nolimits_{ij} z^*_{ij} h_{ij} + \lambda^* \\ \text{s.t. } & h_{ij} \leq h_i, h_j \\ & h_i \in \{0,1\}, \sum\nolimits_i h_i = K \end{aligned}$$

where h_i is a binary variable which indicates if $\mathbf{s}$ covers target i ($h_i = 0$) or not ($h_i = 1$) and $h_{i,j}$ is a binary variable which indicates if $\mathbf{s}$ covers both i and j.

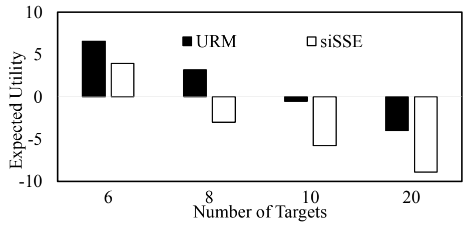
7 Experimental Evaluation

We evaluate the solution quality and runtime of our algorithms on games generated using GAMUT[7]. All our experiments were run on a 2.8GHz Intel Xeon E5-2680v2 processor with 256GB of RAM, using CPLEX 12.8 for solving LP/MILPs. We set the covariance value $r \in [0.0, 1.0]$ in GAMUT with step size $\lambda = 0.2$ to control the correlation of attacker and defender payoffs. All results are averaged over 120 instances (20 games per covariance value). All comparison results with our algorithms are statistically significant under bootstrap-t ($\alpha = 0.05$).

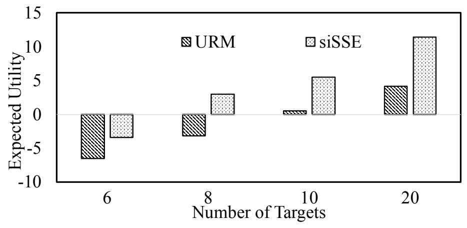
Algorithms and Baselines We show simulation results for two algorithms: (i) *sequential attacks with resource movement* (or `URM`); and (ii) *sequential attacks with no resource movement* (or `NRM`). In addition, we use `siSSE` as a baseline in order to show that the defender can suffer arbitrary losses if he does not take into account sequential attacks in his resource allocation problem. We test all our algorithms and the baseline against a sequential move attacker.

Figures 1a and 1b show solution qualities (i.e., expected game utilities) in the *resource-movement* setting for the defender and attacker (respectively), whereas Figures 2a, 3a and 2b, 3b show the same solution qualities in the *no-resource-movement* setting. The x-axis in Figures 1 and 2 shows increasing number of targets. The x-axis in Figure 3 shows increasing number of resources. The y-axis in all these figures show the expected utility for defender and attacker.

[7] See http://gamut.stanford.edu/.

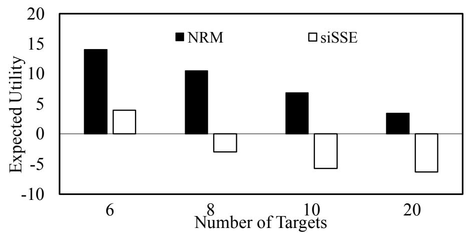


(a) Defender Utility

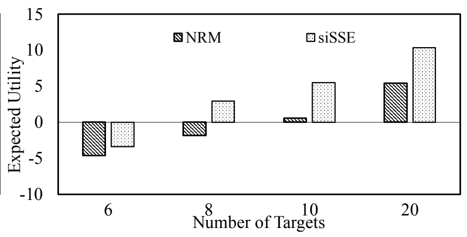


(b) Attacker Utility

Fig. 1: Difference in the utility of the players with increasing number of targets in the resource-movement setting.



(a) Defender Utility



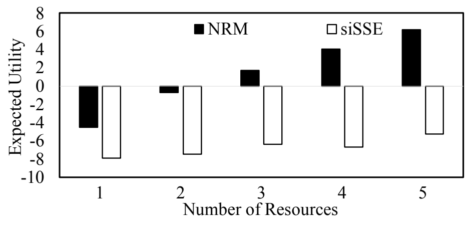
(b) Attacker Utility

Fig. 2: Difference in the utility of the players with increasing number of targets in the no-resource-movement setting.

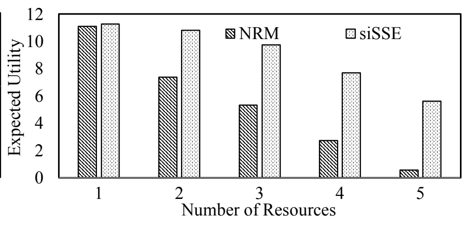
Scaling up Number of Targets Figures 1a and 2a show that against sequential attacks, the expected defender utility achieved by our algorithms (i.e., `URM` and `NRM`) is significantly higher than that achieved by `siSSE` in both resource-movement and no-resource-movement settings. On the other hand, Figures 1b and 2b show that the attacker achieves significantly lower utility with our algorithms as compared to `siSSE`. This shows the importance of taking into account sequential attacks in the defender's optimization problem, and shows that our algorithms are able to successfully outperform the baseline.

Scaling up Number of Resources Next, we show how the solution quality of our algorithms varies with increasing number of resources. The number of targets is set to 10. Figures 3a and 3b show that in the no-resource-movement setting, the defender solution quality increases with increasing number of security resources, whereas the attacker solution quality decreases.

Runtime Results Figures 4a and 4b shows the runtime of our algorithms with increasing number of targets and resources (respectively). The x-axis shows increasing number of targets (resources), and the y-axis shows the runtime (in seconds). These figures show that our `NRM` algorithm runs significantly slower than our `URM`. This makes sense since the `NRM` algorithm needs to solve multiple MILPs and LPs to find cutting planes until it reaches the optimal solution, which is time consuming compared to the `URM` algorithm. As expected, the `siSSE`

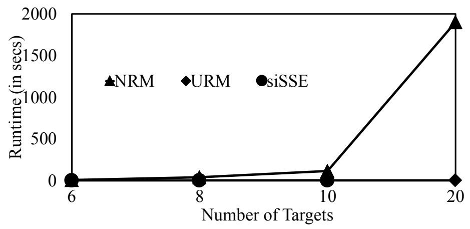


(a) Defender Utility

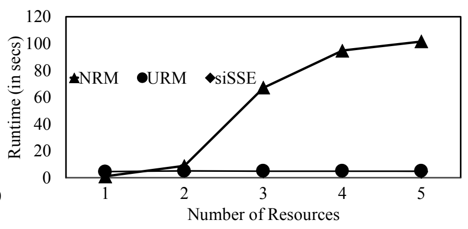


(b) Attacker Utility

Fig. 3: Difference in the utility of the players with increasing number of resources in the no-resource movement setting.

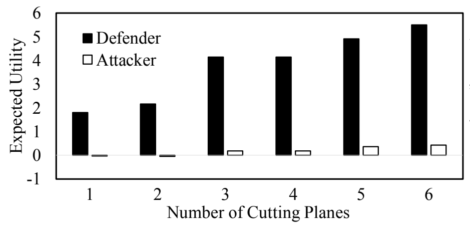


(a) Scaling up the number of targets.

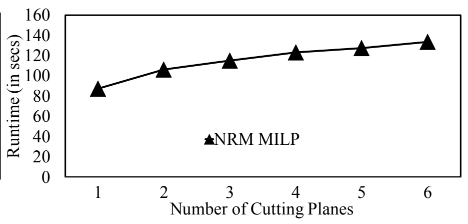


(b) Scaling up the number of resources.

Fig. 4: Comparison of the computation time.



(a) Solution Quality



(b) Runtime

Fig. 5: The impact of cutting planes on solution quality and runtime performance in the no-resource-movement setting

algorithm runs quicker than both our algorithms, but as shown in Figures 1, 2 and 3, it performs significantly worse than our algorithms in terms of solution quality. This establishes the superiority of our algorithms performance over the state-of-the-art baseline in tackling sequential attacks.

Finally, we analyze the impact of limiting the maximum number of cutting planes that the `NRM` algorithm can generate on its runtime and solution quality. Figure 5a and 5b show the variation in solution quality and runtime (respectively) of the `NRM` algorithm with increasing limits on the number of cutting planes that can be added to the MILP. The number of targets is fixed to 10.

The x-axis shows increasing number of cutting planes and the y-axis shows the solution quality (and runtime). These figures show that beyond three cutting planes, the solution quality of `NRM` shows diminishing returns with higher values of cutting planes, whereas the running time of the algorithm increases at a roughly linear rate with increasing number of cutting planes. This suggests that in practice, `NRM` can be run by setting a limit on the number of cutting planes, beyond which there are only marginal increases in the solution quality.

8 Summary

This paper studies the security problem in which the attacker can attack multiple targets in a sequential manner. In this paper, we introduce a new sequential-attack game model (built upon the Stackelberg game model), which incorporates real-time observations, the behavior of sequential attacks, and strategic plans of non-myopic players. We then propose practical game-theoretic algorithms for computing an equilibrium in different game settings. Our new algorithms exploit intrinsic properties of the equilibrium to derive compact representations of both state history and strategy spaces of players (which are exponential in number in the original representations). Finally, our computational experiments show that the defender and attacker receive significant loss and benefit respectively if the defender does not address sequential attacks. By taking into account sequential attacks, such loss and benefit is reduced drastically.

Acknowledgment This work was supported in part by by the Czech Science Foundation (no. 19-24384Y).

References

1. Basilico, N., Gatti, N., Amigoni, F.: Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In: AAMAS. pp. 57–64 (2009)
2. Bošanský, B., Čermák, J.: Sequence-Form Algorithm for Computing Stackelberg Equilibria in Extensive-Form Games. In: AAAI Conference on Artificial Intelligence (2015)
3. Fang, F., Nguyen, T.H., Pickles, R., Lam, W.Y., Clements, G.R., An, B., Singh, A., Tambe, M., Lemieux, A.: Deploying PAWS: Field optimization of the protection assistant for wildlife security. In: IAAI (2016)
4. Fang, F., Stone, P., Tambe, M.: When Security Games Go Green: Designing Defender Strategies to Prevent Poaching and Illegal Fishing. In: In Proceedings of 24th International Joint Conference on Artificial Intelligence (IJCAI) (2015)
5. Jiang, A.X., Yin, Z., Zhang, C., Tambe, M., Kraus, S.: Game-theoretic Randomization for Security Patrolling with Dynamic Execution Uncertainty. In: Proceedings of 12th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). pp. 207–214 (2013)
6. Karwowski, J., Madziuk, J.: Stackelberg Equilibrium Approximation in General-Sum Extensive-Form Games with Double-Oracle Sampling Method. In: Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019). pp. 2045–2047 (2019)

7. Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Ordóñez, F., Tambe, M.: Computing optimal randomized resource allocations for massive security games. In: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems. pp. 689–696 (2009), http://portal.acm.org/citation.cfm?id=1558013.1558108
8. Korzhyk, D., Conitzer, V., Parr, R.: Security Games with Multiple Attacker Resources. In: IJCAI (2011)
9. Letchford, J., Conitzer, V.: Computing optimal strategies to commit to in extensive-form games. In: Proceedings of the 11th ACM conference on Electronic commerce. pp. 83–92. ACM, New York, NY, USA (2010). https://doi.org/http://doi.acm.org/10.1145/1807342.1807354, http://doi.acm.org/10.1145/1807342.1807354
10. Letchford, J., Vorobeychik, Y.: Computing randomized security strategies in networked domains. Applied Adversarial Reasoning and Risk Modeling **11**, 06 (2011)
11. Lisý, V., Davis, T., Bowling, M.: Counterfactual Regret Minimization in Sequential Security Games. In: Proceedings of AAAI Conference on Artificial Intelligence (2016)
12. Nguyen, T.H., Sinha, A., Gholami, S., Plumptre, A., Joppa, L., Tambe, M., Driciru, M., Wanyama, F., Rwetsiba, A., Critchlow, R., Beale, C.M.: CAPTURE: A New Predictive Anti-Poaching Tool for Wildlife Protection. In: Proceedings of the 2016 International Conference on Autonomous Agents and Multiagent Systems. pp. 767–775. AAMAS, Richland, SC (2016), http://dl.acm.org/citation.cfm?id=2937029.2937037
13. Shieh, E., An, B., Yang, R., Tambe, M., Baldwin, C., DiRenzo, J., Maule, B., Meyer, G.: PROTECT: A deployed game theoretic system to protect the ports of the United States. In: AAMAS (2012)
14. Sinha, A., Fang, F., An, B., Kiekintveld, C., Tambe, M.: Stackelberg security games: Looking beyond a decade of success. In: IJCAI. pp. 5494–5501 (2018)
15. Tambe, M. (ed.): Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned. Cambridge University Press (2011)
16. Čermák, J., Bošanský, B., Durkota, K., Lisý, V., Kiekintveld, C.: Using Correlated Strategies for Computing Stackelberg Equilibria in Extensive-Form Games. In: Proceedings of AAAI Conference on Artificial Intelligence. pp. 439–445 (2016)
17. Černý, J., Bošanský, B., Kiekintveld, C.: Incremental Strategy Generation for Stackelberg Equilibria in Extensive-Form Games. In: Proceedings of ACM Conference on Economics and Computation (EC). pp. 151–168 (2018)
18. Wang, S., Liu, F., Shroff, N.: Non-Additive Security Games. In: Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17). pp. 728–735 (2017)
19. Yin, Z., Jiang, A.X., Johnson, M.P., Tambe, M., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., Sullivan, J.P.: TRUSTS: Scheduling Randomized Patrols for Fare Inspection in Transit Systems. In: Proceedings of 24th Conference on Innovative Applications of Artificial Intelligence (IAAI) (2012)