Chapter 2
Security Games Applied to Real-World: Research Contributions and Challenges

Manish Jain, Bo An, and Milind Tambe

Abstract The goal of this chapter is to introduce a challenging real-world problem for researchers in multiagent systems and beyond, where our collective efforts may have a significant impact on activities in the real-world. The challenge is in applying game theory for security: our goal is not only to introduce the problem, but also to provide exemplars of initial successes of deployed systems in this problem arena. Furthermore, we present key ideas and algorithms for solving and understanding the characteristics large-scale real-world security games, and then present some key open research challenges in this area.

2.1 Introduction

Security is a critical concern around the world that arises in protecting our ports, airports, transportation or other critical national infrastructure from adversaries, in protecting our wildlife and forests from poachers and smugglers, and in curtailing the illegal flow of weapons, drugs and money; and it arises in problems ranging from physical to cyber-physical systems. In all of these problems, we have limited security resources which prevent full security coverage at all times; instead, limited security resources must be deployed intelligently taking into account differences in priorities of targets requiring security coverage, the responses of the attackers to the security posture and potential uncertainty over the types, capabilities, knowledge and priorities of attackers faced.

Game theory is well-suited to adversarial reasoning for security resource allocation and scheduling problems. Casting the problem as a Bayesian Stackelberg game, new algorithms have been developed for efficiently solving such games that

M. Jain (✉) • B. An • M. Tambe
Computer Science Department, University of Southern California, Los Angeles, California 90089, USA
e-mail: manish.jain@usc.edu; boa@usc.edu; tambe@usc.edu

S. Jajodia et al. (eds.), *Moving Target Defense II: Application of Game Theory and Adversarial Modeling*, Advances in Information Security 100,
DOI 10.1007/978-1-4614-5416-8_2,

provide randomized patrolling or inspection strategies. These algorithms have led to some initial successes in this challenge problem arena, leading to advances over previous approaches in security scheduling and allocation, e.g., by addressing key weaknesses of predictability of human schedulers. These algorithms are now deployed in multiple applications: ARMOR has been deployed at the Los Angeles International Airport (LAX) since 2007 to randomize checkpoints on the roadways entering the airport and canine patrol routes within the airport terminals [1]; IRIS, a game-theoretic scheduler for randomized deployment of the US Federal Air Marshals (FAMS) requiring significant scale-up in underlying algorithms, has been in use since 2009 [2]; PROTECT, which uses a new set of algorithms based on quantal-response is deployed in the port of Boston for randomizing US coast guard patrolling [3,4]; PROTECT has been deployed in the port of Boston since April 2011 and is now in use at the port of New York; GUARDS is under evaluation for national deployment by the US Transportation Security Administration (TSA) [5], and TRUSTS is being tested by the Los Angeles Sheriffs Department (LASD) in the LA Metro system to schedule randomized patrols for fare inspection [6]. These initial successes point the way to major future applications in a wide range of security arenas; with major research challenges in scaling up our game-theoretic algorithms, to addressing human adversaries' bounded rationality and uncertainties in action execution and observation, as well as in preference elicitation and multiagent learning.

This paper will provide an overview of the models and algorithms, key research challenges and a brief description of our successful deployments. While initial research has made a start, a lot remains to be done; yet these are large-scale interdisciplinary research challenges that call upon multiagent researchers to work with researchers in other disciplines, be "on the ground" with domain experts, and examine real-world constraints and challenges that cannot be abstracted away.

2.2 Stackelberg Security Games

Security problems are increasingly studied using Stackelberg games, since Stackelberg games can appropriately model the strategic interaction between a defender and an attacker. Stackelberg games were first introduced to model leadership and commitment [7], and are now widely used to study security problems ranging from "police and robbers" scenario [8], computer network security [9], missile defense systems [10], and terrorism [11]. Models for arms inspections and border patrolling have also been modeled using inspection games [12], a related family of Stackelberg games.

The wide use of Stackelberg games has inspired theoretic and algorithmic progress leading to the development of fielded applications. These algorithms are central to many fielded applications, as described in Sect. 2.3. For example, DOBSS [13], an algorithm for solving Bayesian Stackelberg games, is central to a fielded application in use at the Los Angeles International Airport [1]. Similarly,

Table 2.1 Example security game with two targets

	Defender		Attacker	
Target	Covered	Uncovered	Covered	Uncovered
t_1	10	0	−1	1
t_2	0	−10	−1	1

Conitzer and Sandholm [14] give complexity results and algorithms for computing optimal commitment strategies in Bayesian Stackelberg games, including both pure and mixed-strategy commitments. This chapter provides details on this use of Stackelberg games for modeling security domains. We first give a generic description of security domains followed by *security games*, the model by which security domains are formulated in the Stackelberg game framework.

2.2.1 Security Domains

In a security domain, a defender must perpetually defend a set of targets using a limited number of resources, whereas the attacker is able to surveil and learn the defender's strategy and attacks after careful planning. This fits precisely into the description of a Stackelberg game if we map the defender to the leader's role and the attacker to the follower's role [12, 15]. An action, or *pure strategy*, for the defender represents deploying a set of resources on patrols or checkpoints, e.g. scheduling checkpoints at the LAX airport or assigning federal air marshals to protect flight tours. The pure strategy for an attacker represents an attack at a target, e.g., a flight. The strategy for the leader is a mixed strategy, a probability distribution over the pure strategies of the defender. Additionally, with each target are also associated a set of payoff values that define the utilities for both the defender and the attacker in case of a successful or a failed attack. These payoffs are represented using the *security game* model, described next.

2.2.2 Security Games

In a security game, a set of four payoffs is associated with each target. These four payoffs are the reward and penalty to both the defender and the attacker in case of a successful or an unsuccessful attack, and are sufficient to define the utilities for both players for all possible outcomes in the security domain. Table 2.1 shows an example security game with two targets, t_1 and t_2. In this example game, if the defender was *covering* (protecting) target t_1 and the attacker attacked t_1, the defender would get 10 units of reward whereas the attacker would receive −1 units.

Security games make the assumption that it is always better for the defender to cover a target as compared to leaving it uncovered, whereas it is always better for the attacker to attack an uncovered target. This assumption is consistent with the

Table 2.2 Example Bayesian security game with two targets and two attacker types

	Attacker Type 1					Attacker Type 2			
	Defender		Attacker			Defender		Attacker	
Target	Cov.	Uncov.	Cov.	Uncov.	Target	Cov.	Uncov.	Cov.	Uncov.
t_1	10	0	−1	1	t_1	5	−4	−2	1
t_2	0	−10	−1	1	t_2	4	−5	−1	2

payoff trends in the real-world. Another crucial feature of the security games is that the payoff of an outcome depends only on the target attacked, and whether or not it is covered by the defender [16]. The payoffs do *not* depend on the remaining aspects of the defender allocation. For example, if an adversary succeeds in attacking target t_1, the penalty for the defender is the same whether the defender was guarding target t_2 or not. Therefore, from a payoff perspective, many resource allocations by the defender are identical. This is exploited during the computation of a defender strategy: only the coverage probability of each target is required to compute the utilities of the defender and the attacker.

The Bayesian extension to the Stackelberg game allows for multiple types of players, with each associated with its own payoff values [13, 17]. Bayesian games are used to model uncertainty over the payoffs and preferences of the players; indeed more uncertainty can be expressed with increasing number of types. For the security games of interest, there is only one leader type (e.g., only one police force), although there can be multiple follower types (e.g., multiple attacker types trying to infiltrate security). Each follower type is represented using a different payoff matrix, as shown by an example with two attacker types in Table 2.2. The leader does not know the follower's type, but knows the probability distribution over them. The goal is to find the optimal mixed strategy for the leader to commit to, given that the defender could be facing any of the follower types.

2.2.3 Solution Concept: Strong Stackelberg Equilibrium

The solution to a security game is a mixed strategy for the defender that maximizes the expected utility of the defender, given that the attacker learns the mixed strategy of the defender and chooses a best-response for himself. This solution concept is known as a Stackelberg equilibrium [18]. However, the solution concept of choice in all deployed applications is a *strong* form of the Stackelberg equilibrium [19], which assumes that the follower will always break ties in favor of the leader in cases of indifference. This is because a strong Stackelberg equilibrium (SSE) exists in all Stackelberg games, and additionally, the leader can always induce the favorable strong equilibrium by selecting a strategy arbitrarily close to the equilibrium that causes the follower to strictly prefer the desired strategy [20]. Indeed, SSE is the mostly commonly adopted concept in related literature [13, 14, 21].

A SSE for security games is informally defined as follows (the formal definition of SSE is not introduced for brevity, and can instead be found in [16]):

Definition 2.1. A pair of strategies form a *Strong Stackelberg Equilibrium* (SSE) if they satisfy:

1. The defender plays a best-response, that is, the defender cannot get a higher payoff by choosing any other strategy.
2. The attacker play a best-response, that is, given a defender strategy, the attacker cannot get a higher payoff by attacking any other target.
3. The attacker breaks ties in favor of the leader.

2.3 Deployed and Emerging Security Applications

We now talk about five successfully deployed applications that use the concept of strong Stackelberg Equilibrium to suggest security scheduling strategies to the defender in different real-world domains.

2.3.1 ARMOR *for Los Angeles International Airport*

Los Angeles International Airport (LAX) is the largest destination airport in the United States and serves 60–70 million passengers per year. The LAX police use diverse measures to protect the airport, which include vehicular checkpoints, police units patrolling the roads to the terminals, patrolling inside the terminals (with canines), and security screening and bag checks for passengers. The application of game-theoretic approach is focused on two of these measures: (1) placing vehicle checkpoints on inbound roads that service the LAX terminals, including both location and timing, and (2) scheduling patrols for bomb-sniffing canine units at the different LAX terminals. The eight different terminals at LAX have very different characteristics, like physical size, passenger loads, foot traffic or international versus domestic flights. These factors contribute to the differing risk assessments of these eight terminals. Furthermore, the numbers of available vehicle checkpoints and canine units are limited by resource constraints. Thus, it is challenging to optimally allocate these resources to improve their effectiveness while avoiding patterns in the scheduled deployments.

The ARMOR system (Assistant for Randomized Monitoring over Routes) focuses on two of the security measures at LAX (checkpoints and canine patrols) and optimizes security resource allocation using Bayesian Stackelberg games. Take the vehicle checkpoints model as an example. Assume that there are n roads, the police's strategy is placing $m < n$ checkpoints on these roads where m is the maximum number of checkpoints. The adversary may potentially choose to attack through one of these roads. ARMOR models different types of attackers with different

payoff functions, representing different capabilities and preferences for the attacker. ARMOR uses DOBSS (Decomposed Optimal Bayesian Stackelberg Solver) [13] to compute the defender's optimal strategy. ARMOR has been successfully deployed since August 2007 at LAX [1, 22].

2.3.2 IRIS *for US Federal Air Marshals Service*

The US Federal Air Marshals Service (FAMS) allocates air marshals to flights originating in and departing from the United States to dissuade potential aggressors and prevent an attack should one occur. Flights are of different importance based on a variety of factors such as the numbers of passengers, the population of source/destination, international flights from different countries, and special events that can change the risks for particular flights at certain times. Security resource allocation in this domain is significantly more challenging than for ARMOR: a limited number of air marshals need to be scheduled to cover thousands of commercial flights each day. Furthermore, these air marshals must be scheduled on tours of flights that obey various constraints (e.g., the time required to board, fly, and disembark). Simply finding schedules for the marshals that meet all of these constraints is a computational challenge. Our task is made more difficult by the need to find a randomized policy that meets these scheduling constraints, while also accounting for the different values of each flight.

Against this background, the IRIS system (Intelligent Randomization In Scheduling) has been developed and has been deployed by FAMS since October 2009 to randomize schedules of air marshals on international flights. In IRIS, the targets are the set of n flights and the attacker could potentially choose to attack one of these flights. The FAMS can assign $m < n$ air marshals that may be assigned to protect these flights.

Since the number of possible schedules exponentially increases with the number of flights and resources, DOBSS is no longer applicable to the FAMS domain. Instead, IRIS uses the much faster ASPEN algorithm [23] to generate the schedule for thousands of commercial flights per day. IRIS also uses an attribute-based preference elicitation system to determine reward values for the Stackelberg game model.

2.3.3 PROTECT *for US Coast Guard*

The US Coast Guard's (USCG) mission includes maritime security of the US coasts, ports, and inland waterways; a security domain that faces increased risks due to threats such as terrorism and drug trafficking. Given a particular port and the variety of critical infrastructure that an attacker may attack within the port, USCG conducts patrols to protect this infrastructure; however, while the attacker has the opportunity to observe patrol patterns, limited security resources imply that USCG patrols

Fig. 2.1 USCG boats patrolling the ports of Boston and NY. (**a**) PROTECT is being used in Boston, (**b**) Extending PROTECT to NY

cannot be at every location 24/7. To assist the USCG in allocating its patrolling resources, the PROTECT (Port Resilience Operational/Tactical Enforcement to Combat Terrorism) model has been designed to enhance maritime security. It has been in use at the port of Boston since April 2011, and now is also in use at the port of New York (Fig. 2.1). Similar to previous applications ARMOR and IRIS, PROTECT uses an attacker–defender Stackelberg game framework, with USCG as the defender against terrorists that conduct surveillance before potentially launching an attack.

The goal of PROTECT is to use game theory to assist the USCG in maximizing its effectiveness in the Ports, Waterways, and Coastal Security (PWCS) Mission. PWCS patrols are focused on protecting critical infrastructure; without the resources to provide 100 percent on scene presence at any, let alone all, of the critical infrastructure, optimization of security resource is critical. Towards that end, unpredictability creates situations of uncertainty for an enemy and can be enough to deem a target less appealing. The PROTECT system, focused on the PWCS patrols, addresses how the USCG should optimally patrol critical infrastructure in a port to maximize protection, knowing that the attacker may conduct surveillance and then launch an attack. While randomizing patrol patterns is key, PROTECT also addresses the fact that the targets are of unequal value, understanding that the attacker will adapt to whatever patrol patterns USCG conducts. The output of PROTECT is a schedule of patrols which includes when the patrols are to begin, what critical infrastructure to visit for each patrol, and what activities to perform at each critical infrastructure.

While PROTECT builds on previous work, it offers some key innovations. First, this system is a departure from the assumption of perfect attacker rationality noted in previous work, relying instead on a quantal response (QR) model [24] of the attacker's behavior. Second, to improve PROTECT's efficiency, a compact representation of the defender's strategy space is used by exploiting equivalence and dominance. Finally, the evaluation of PROTECT for the first time provides real-world data: (a) comparison of human-generated vs PROTECT security

schedules, and (b) results from an Adversarial Perspective Team's (human mock attackers) analysis. The PROTECT model is now being extended to the port of New York and it may potentially be extended to other ports in the US.

2.3.4 GUARDS *for US Transportation Security Agency*

The United States Transportation Security Administration (TSA) is tasked with protecting the nation's over 400 airports which services approximately 28,000 commercial flights and up to approximately 87,000 total flights per day. To protect this large transportation network, the TSA employs approximately 48,000 Transportation Security Officers, who are responsible for implementing security activities at each individual airport. While many people are aware of common security activities, such as individual passenger screening, this is just one of many security layers TSA personnel implement to help prevent potential threats [25, 26]. These layers can involve hundreds of heterogeneous security activities executed by limited TSA personnel leading to a complex resource allocation challenge. While activities like passenger screening are performed for every passenger, the TSA cannot possibly run every security activity all the time. Thus, while the resources required for passenger screening are always allocated by the TSA, it must also decide how to appropriately allocate its remaining security officers among the layers of security to protect against a number of potential threats, while facing challenges such as surveillance and an adaptive attacker as mentioned before.

To aid the TSA in scheduling resources to protect airports, a new application called GUARDS (Game-theoretic Unpredictable and Randomly Deployed Security) has been developed. While GUARDS also utilizes Stackelberg games as ARMOR and IRIS, GUARDS faces three key challenges [5]: (1) reasoning about hundreds of heterogeneous security activities; (2) reasoning over diverse potential threats; and (3) developing a system designed for hundreds of end-users. To address those challenges, GUARDS created a new game-theoretic framework that allows for heterogeneous defender activities and compact modeling of a large number of threats and developed an efficient solution technique based on general-purpose Stackelberg game solvers. GUARDS is currently under evaluation and testing for scheduling practices at an undisclosed airport. If successful, the TSA intends to incorporate the system into their unpredictable scheduling practices nationwide.

2.3.5 TRUSTS *for Urban Security in Transit Systems*

In some urban transit systems, including the Los Angeles Metro Rail system, passengers are legally required to purchase tickets before entering but are not physically forced to do so (Fig. 2.2). Instead, security personnel are dynamically deployed throughout the transit system, randomly inspecting passenger tickets.

Fig. 2.2 TRUSTS for transit systems. (**a**) Los Angeles Metro, (**b**) Barrier-free entrance to transit system

This proof-of-payment fare collection method is typically chosen as a more cost-effective alternative to direct fare collection, i.e., when the revenue lost to fare evasion is believed to be less than what it would cost to directly preclude it.

Take the Los Angeles Metro as an example. With approximately 300,000 riders daily, this revenue loss can be significant; the annual cost has been estimated at $5.6 million [27]. The Los Angeles Sheriffs Department (LASD) deploys uniformed patrols on board trains and at stations for fare-checking (and for other purposes such as crime prevention), in order to discourage fare evasion. With limited resources to devote to patrols, it is impossible to cover all locations at all times. The LASD thus requires some mechanism for choosing times and locations for inspections. Any predictable patterns in such a patrol schedule are likely to be observed and exploited by potential fare-evaders. The LASD's current approach relies on humans for scheduling the patrols. However, human schedulers are poor at generating unpredictable schedules; furthermore such scheduling for LASD is a tremendous cognitive burden on the human schedulers who must take into account all of the scheduling complexities (e.g., train timings, switching time between trains, and schedule lengths).

The TRUSTS system (Tactical Randomization for Urban Security in Transit Systems) models the patrolling problem as a leader–follower Stackelberg game [28]. The leader (LASD) pre-commits to a mixed strategy patrol (a probability distribution over all pure strategies), and riders observe this mixed strategy before deciding whether to buy the ticket or not. Both ticket sales and fines issued for fare evasion translate into revenue for the government. Therefore the optimization objective for the leader is to maximize total revenue (total ticket sales plus penalties). Urban transit systems, however, present unique computational challenges since there are exponentially many possible patrol strategies, each subject to both the spatial and temporal constraints of travel within the transit network under consideration. To overcome this challenge, TRUSTS uses a compact representation which captures

Fig. 2.3 The terrorist attacks of 2008 in Mumbai

the spatial as well as temporal structure of the domain. The LASD is currently testing TRUSTS in the LA Metro system by deploying patrols according to the generated schedules and measuring the revenue recovered.

2.3.6 Future Applications

Beyond the deployed and emerging applications above are a number of different application areas. One such area of great importance is securing urban city networks, transportation networks, computer networks and other network centric security domains. For example, after the terrorist attacks in Mumbai of 2008 [29] (refer Fig. 2.3), the Mumbai police have started setting up vehicular checkpoints on roads. We can model the problem faced by the Mumbai police as a security game between the Mumbai police and an attacker. In this urban security game, the pure strategies of the defender correspond to allocations of resources to edges in the network—for example, an allocation of police checkpoints to roads in the city. The pure strategies of the attacker correspond to paths from any *source* node to any *target* node—for example, a path from a landing spot on the coast to the airport.

Another area is protecting forests [30], where we must protect a continuous forest area from extractors by patrols through the forest that seek to deter such extraction activity. With limited resources for performing such patrols, a patrol strategy will seek to distribute the patrols throughout the forest, in space and time, in order to minimize the resulting amount of extraction that occurs or maximize the degree of forest protection. This problem can be formulated as a Stackelberg game and the focus is on computing optimal allocations of patrol density [30].

Another potential application is police patrols for crime suppression which is a data-intensive domain [31]. Thus, it would be promising to use data mining tools on a database of reported crimes and other events to identify the locations to be patrolled, the times at which the game changes, and the types of attackers faced. The idea is to exploit temporal and spatial patterns of crime in the area to be patrolled to determine the priorities on how to use the limited security resources. However, even with all of these applications, we have barely scratched the surface of possibilities in terms of potential applications for multiagent researchers for applying game theory for security.

The Stackelberg game framework can also be applied to adversarial domains that exhibit "contagious" actions for each player. For example, word-of-mouth advertising/viral marketing has been widely studied by marketers trying to understand why one product or video goes "viral" while others go unnoticed [32]. Counter-insurgency is the contest for the support of the local leaders in an armed conflict and can include a variety of operations such as providing security and giving medical supplies. Just as in word-of-mouth advertising and peacekeeping operations, these efforts carry a social effect beyond the action taken that can cause advantageous ripples through the neighboring population. Moreover, multiple intelligent parties attempt to leverage the same social network to spread their message, necessitating an adversary-aware approach to strategy generation. Game-theoretic approaches can be used to generate resource allocations strategies for such large-scale, real world networks. This interaction can be modeled as a graph with one player attempting to spread influence while the other player attempts to stop the probabilistic propagation of that influence by spreading their own influence. This "blocking" problem models situations faced by governments/peacekeepers combatting the spread of terrorist radicalism and armed conflict with daily/weekly/monthy visits with local leaders to provide support and discuss grievances [33].

Game-theoretic methods are also appropriate for modeling resource allocation in cybersecurity [34] such as packet selection and inspection for detecting potential threats in large computer networks [35]. The problem of attacks on computer systems and corporate computer networks gets more pressing each year as the sophistication of the attacks increases together with the cost of their prevention. A number of intrusion detection and monitoring systems are being developed, e.g., deep packet inspection method that periodically selects a subset of packets in a computer network for analysis. However, there is a cost associated with the deep packet inspection, as it leads to significant delays in the throughput of the network. Thus, the monitoring system works under a constraint of limited selection of a

fraction of all packets which can be inspected. The attacking/protecting problem can be formulated as a game between two players: the attacker (or the intruder), and the defender (the detection system) [35]. The intruder wants to gain control over (or to disable) a valuable computer in the network by scanning the network, hacking into a more vulnerable system, and/or gaining access to further devices on the computer network. The actions of the attacker can therefore be seen as sending malicious packets from a controlled computer (termed source) to a single or multiple vulnerable computers (termed targets). The objective of the defender is to prevent the intruder from succeeding by selecting the packets for inspection, identifying the attacker, and subsequently thwarting the attack. However, packet inspections cause unwanted latency and hence the defender has to decide where and how to inspect network traffic in order to maximize the probability of a successful malicious packet detection. The computational challenge is efficiently computing the optimal defending strategies for such network scenarios [35].

2.4 Scaling Up To Real-World Problem Sizes

Real world problems, like the FAMS and urban road networks, present billions of pure strategies to both the defender and the attacker. Such large problem instances cannot even be represented in modern computers, let alone solved using previous techniques. We now describe models and algorithms that compute optimal defender strategies for massive real-world security domains. In particular, we describe the following algorithms: (a) ASPEN, an algorithm to compute strong Stackelberg equilibria (SSE) in domains with a *very large* number of pure strategies (up to billions of actions) for the defender [23]; (b) RUGGED, an algorithm to compute the optimal defender strategy in domains with a very large number of pure strategies for both the defender and the attacker [36]; and (c) a hierarchical framework for Bayesian games that is applicable to all Stackelberg solvers, that can be combined with the strategy generation techniques of ASPEN [17]. These algorithms provide scale-ups in real-world domains by efficiently analyzing the strategy space of the players. ASPEN and RUGGED use strategy generation: the algorithms start by considering a minimal set of pure strategies for both the players (defender and attacker). Pure strategies are then generated iteratively, and a strategy is added to the set only if it would help increase the payoff of the corresponding player (a defender's pure strategy is added if it helps increase the defender's payoff). This process is repeated until the optimal solution is obtained. On the other hand, the hierarchical approach pre-processes the Bayesian Stackelberg game and eliminates strategies that can never be the best response of the players. This approach of strategy generation and elimination not only provides the required scale-ups, but also the mathematical guarantees on solution quality. Finally, in this section, we also describe the d:s ratio, an algorithm-independent property of security game instances

Algorithm 1: Strategy generation in ASPEN

```
1. Initialize P                    // initialize, P: set of pure strategies of the defender
2. Solve Master Problem            // compute optimal defender mixed strategy, given P
3. Calculate cost coefficients from solution of master
4. Update objective of slave problem with coefficients
      // compute whether a pure strategy will increase defender's payoff
5. Solve Slave Problem             // generate the pure strategy that is likely to increase
the defender's payoff the most
if Optimal solution obtained then
   6. Return (x, P)
else
   7. Extract new pure strategy and add to P
   8. Repeat from Step 2
```

that influences the hardness of computation for the given problem instance and is robust across domains, domain representations, algorithms and underlying solvers.

2.4.1 Scaling Up with Defender Pure Strategies

In this section, we describe how ASPEN generates pure strategies for the defender in domains where the number of pure strategies of the defender can be prohibitively large. As an example, let us consider the problem faced by the Federal Air Marshals Service (FAMS). There are currently tens of thousands of commercial flights flying each day, and public estimates state that there are thousands of air marshals that are scheduled daily by the FAMS [37]. Air marshals must be scheduled on tours of flights that obey logistical constraints (e.g., the time required to board, fly, and disembark). An example of a valid schedule is an air marshal assigned to a round trip tour from Los Angeles to New York and back.

ASPEN [23] casts this problem as a security game, where the attacker can choose any of the flights to attack, and each air marshal can cover one schedule. Each schedule here is a feasible set of targets that can be covered together; for the FAMS, each schedule would represent a flight tour which satisfies all the logistical constraints that an air marshal could fly. A *joint schedule* then would assign every air marshal to a flight tour, and there could be exponentially many joint schedules in the domain. A pure strategy for the defender in this security game is a joint schedule. As mentioned previously, ASPEN employs strategy generation since all the defender pure strategies (or joint schedules) cannot be enumerated for such a massive problem. ASPEN decomposes the problem into a *master* problem and a *slave* problem, which are then solved iteratively, as described in Algorithm 1. Given a limited number of pure strategies, the master solves for the defender and the attacker optimization constraints, while the slave is used to generate a new pure strategy for the defender in every iteration.

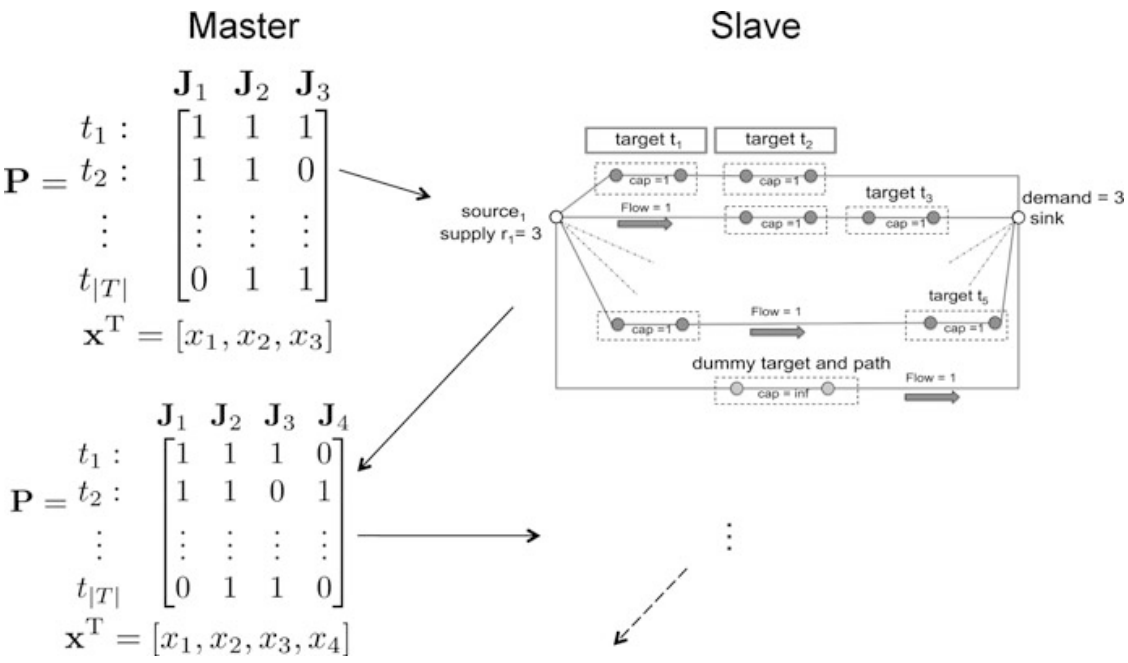

Fig. 2.4 Strategy generation employed in ASPEN: The schedules for a defender are generated iteratively. The *slave* problem is a novel minimum-cost integer flow formulation that computes the new pure strategy to be added to $\mathbf{P}$; $\mathbf{J}_4$ is computed and added in this example

The iteratively process of Algorithm 1 is graphically depicted in Fig. 2.4. The master operates on the pure strategies (joint schedules) generated thus far (Step 2), which are represented using the matrix $\mathbf{P}$. Each column of $\mathbf{P}$, $\mathbf{J}_j$, is one pure strategy (or joint schedule). An entry P_{ij} in the matrix $\mathbf{P}$ is 1 if a target t_i is covered by joint-schedule $\mathbf{J}_j$, and 0 otherwise. The objective of the master problem is to compute $\mathbf{x}$, the optimal mixed strategy of the defender over the pure strategies in $\mathbf{P}$. The objective of the slave problem is to generate the best joint schedule to add to $\mathbf{P}$ (Step 5). The best joint schedule is identified using the concept of *reduced costs* [38] (Step 3–4), which measures if a pure strategy can potentially increase the defender's expected utility (the details of the approach are provided in [23]). While a naïve approach would be to iterate over all possible pure strategies to identify the pure strategy with the maximum potential, ASPEN uses a novel minimum-cost integer flow problem to efficiently identify the best pure strategy to add. ASPEN always converges on the optimal mixed strategy for the defender; the proof can be found in [23].

Employing strategy generation for large optimization problems is not an "out-of-the-box" approach, the problem has to be formulated in a way that allows for domain properties to be exploited. The novel contribution of ASPEN is to provide a linear formulation for the master and a minimum-cost integer flow formulation for the slave, which enable the application of strategy generation techniques. Additionally, ASPEN also provides a branch-and-bound heuristic to reason over attacker actions. This branch-and-bound heuristic provides a further order of magnitude speed-up, allowing ASPEN to handle the massive sizes of real-world problems.

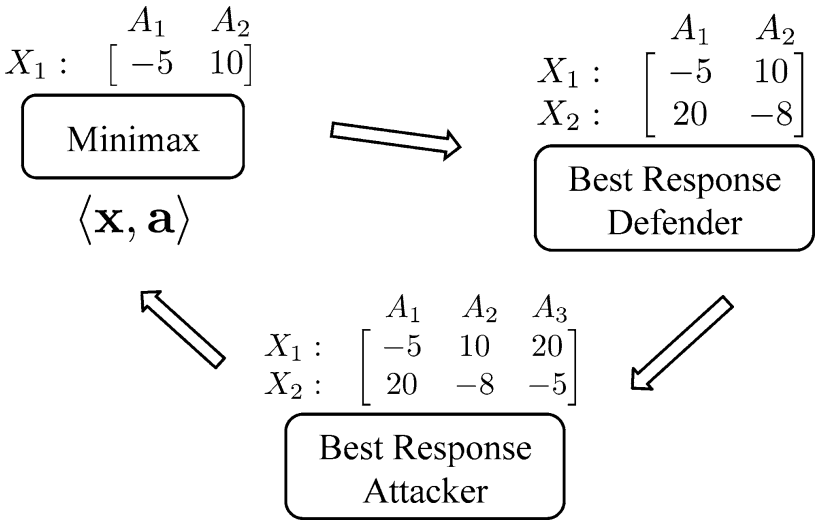

Fig. 2.5 Strategy Generation employed in RUGGED: The pure strategies for both the defender and the attacker are generated iteratively

2.4.2 Scaling Up with Defender and Attacker Pure Strategies

In this section, we describe how RUGGED generates pure strategies for both the defender and the attacker in domains where the number of pure strategies of the players are exponentially large. Let us consider as an example the urban network security game.

Here, strategy generation is required for both the defender and the attacker since the number of pure strategies of both the players are prohibitively large. Figure 2.5 shows the working of RUGGED: here, the minimax module generates the optimal mixed strategies $\langle \mathbf{x}, \mathbf{a} \rangle$ for the two players, whereas the two best response modules generate new strategies for the two players respectively. The rows X_i in the figure are the pure strategies for the defender, they would correspond to an allocation of checkpoints in the urban road network domain. Similarly, the columns A_j are the pure strategies for the attacker, they represent the attack paths in the urban road network domain. The values in the matrix represent the payoffs to the defender. RUGGED models the domain as a zero-sum game, and computes the minimax equilibrium, since the payoff of the minimax strategy is equivalent to the SSE payoff in zero-sum games [39].

The contribution of RUGGED is to provide the mixed integer formulations for the best response modules which enable the application of such a strategy generation approach. RUGGED can compute the optimal solution for deploying up to 4 resources in real-city network with as many as 250 nodes within a reasonable time frame of 10 h (the complexity of this problem can be estimated by observing that both the best response problems are NP-hard themselves [36]). While enhancements to RUGGED are required for deployment in larger real-world domains, RUGGED has opened new possibilities for scaling up to larger games.

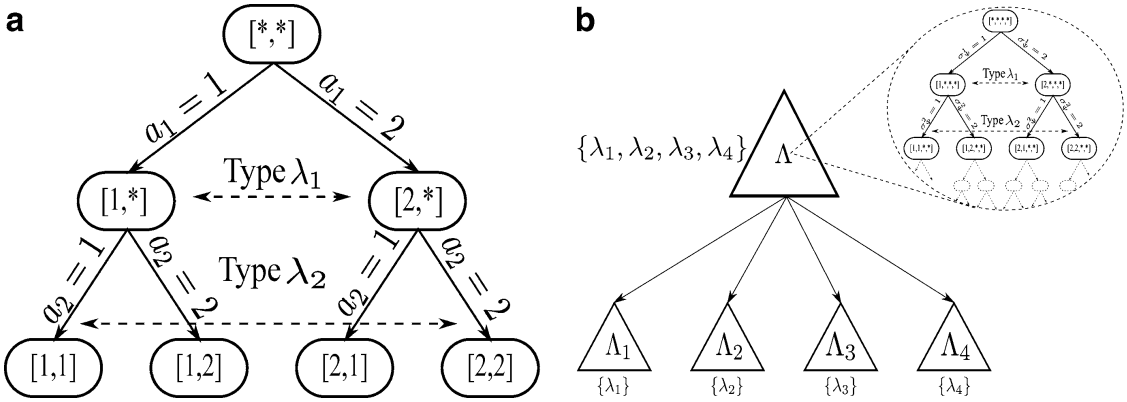

Fig. 2.6 Hierarchical approach for solving Bayesian Stackelberg games. (**a**) Attacker action tree, for a Bayesian game with attackers of two types, with two attacker actions each, (**b**) Hierarchical game tree, decomposing a Bayesian Stackelberg game with 4 types into 4 *restricted* games with one type each

2.4.3 Scaling Up with Attacker Types

The different preferences of different attacker types are modeled through Bayesian Stackelberg games. Computing the optimal leader strategy in Bayesian Stackelberg game is NP-hard [14], and polynomial time algorithms cannot achieve approximation ratios better than $O(types)$ [40]. We now describe HBGS, a hierarchical technique for solving large Bayesian Stackelberg games that decomposes the entire game into many hierarchically-organized *restricted* Bayesian Stackelberg games; it then utilizes the solutions of these restricted games to more efficiently solve the larger Bayesian Stackelberg game [17].

Figure 2.6a shows the *attacker action tree* (a tree depicting all possible pure strategies of an attacker in a Bayesian game, arranged as a tree) of an example Bayesian Stackelberg game with 2 types and 2 actions per attacker type. The leaf nodes of this tree are all the possible action combinations for the attacker in this game, for example, leaf $[1,1]$ implies that the attackers of both types λ_1 and λ_2 chose action a_1. All the leaves of this tree (i.e. all combinations of attacker strategies for all attacker types) need to be evaluated before the optimal strategy for the defender can be computed. This tree is typically evaluated using branch-and-bound. This requirement to evaluate the exponential number of leaves (or pure strategy combinations) is the cause of NP-hardness of Bayesian Stackelberg games; indeed the performance of algorithms can be improved if leaves can be pruned by pre-processing.

The overarching idea of hierarchical structure is to improve the performance of branch-and-bound on the attacker action tree (Fig. 2.6a) by pruning leaves of this tree. It decomposes the Bayesian Stackelberg game into many hierarchically-organized smaller games, as shown by an example in Fig. 2.6b. Each of the restricted games ("child" nodes in Fig. 2.6b) consider only a few attacker types, and are thus exponentially smaller that the Bayesian Stackelberg game at the "parent".

For instance, in this example, the Bayesian Stackelberg game has 4 types ($\langle \lambda_1, \lambda_2, \lambda_3, \lambda_4 \rangle$) and it is decomposed into 4 *restricted* games (leaf nodes) with each restricted game having exactly 1 attacker type. The solutions obtained for the restricted games at the child nodes of the hierarchical game tree are used to provide: (a) pruning rules, (b) tighter bounds, and (c) efficient branching heuristics to solve the bigger game at the parent node faster.

Such hierarchical techniques have seen little application towards obtaining optimal solutions in Bayesian games, while Stackelberg settings have not seen any application of such hierarchical decomposition. This hierarchical idea of HBGS has also been combined with the strategy generation of ASPEN by the HBSA algorithm [17], which is now opening new possibilities to handle uncertainty in large games.

2.4.4 Characterizing Hard Security Game Problem Instances

In this section, we will focus on the runtime required by the different algorithms that compute solutions for instances of security games for three of the domains described above: the LAX or ARMOR domain, the FAMS or IRIS domain and the Coast Guard or PROTECT domain. We will describe the d:s ratio, a domain-spanning measure of the density of defender coverage in any security problem. We will then present results that show that the computationally hardest problem instances of security games occur when this ratio is 0.5.

However, first lets look at the runtime required by the different algorithms that compute solutions for different security domains. Figure 2.7 shows the runtime for computing solutions using DOBSS, ASPEN and a multiple-attacker branch-and-price algorithm for the ARMOR, IRIS and PROTECT[1] domains respectively. The x-axis in each figure shows the number of available resources to the defender, and the y-axis shows the runtime in seconds. In the ARMOR and the IRIS domains, we define the number of resources as the number of officers available to the defender; in the PROTECT domain, we define it as the maximum feasible tour length. These graphs show that there is no unified value of the number of defender resources which makes security game instances hard to solve. Even normalizing the number of resources by the number of targets shows similar inconsistencies across different domains.

We now describe the concept of the *deployment-to-saturation* (d:s) ratio [41], a concept that unifies the domain independent properties of problem instances across different security domains. Specifically, this concept of d:s ratio has been applied to the three security domains described above, eight different MIP algorithms

[1]The deployed application of PROTECT uses a quantal response model whereas the multiple attacker branch-and-price algorithm computes SSE as originally defined [41]. Here, by the PROTECT domain, we refer to the security domain where defender has to execute patrols that are bounded by their tour length.

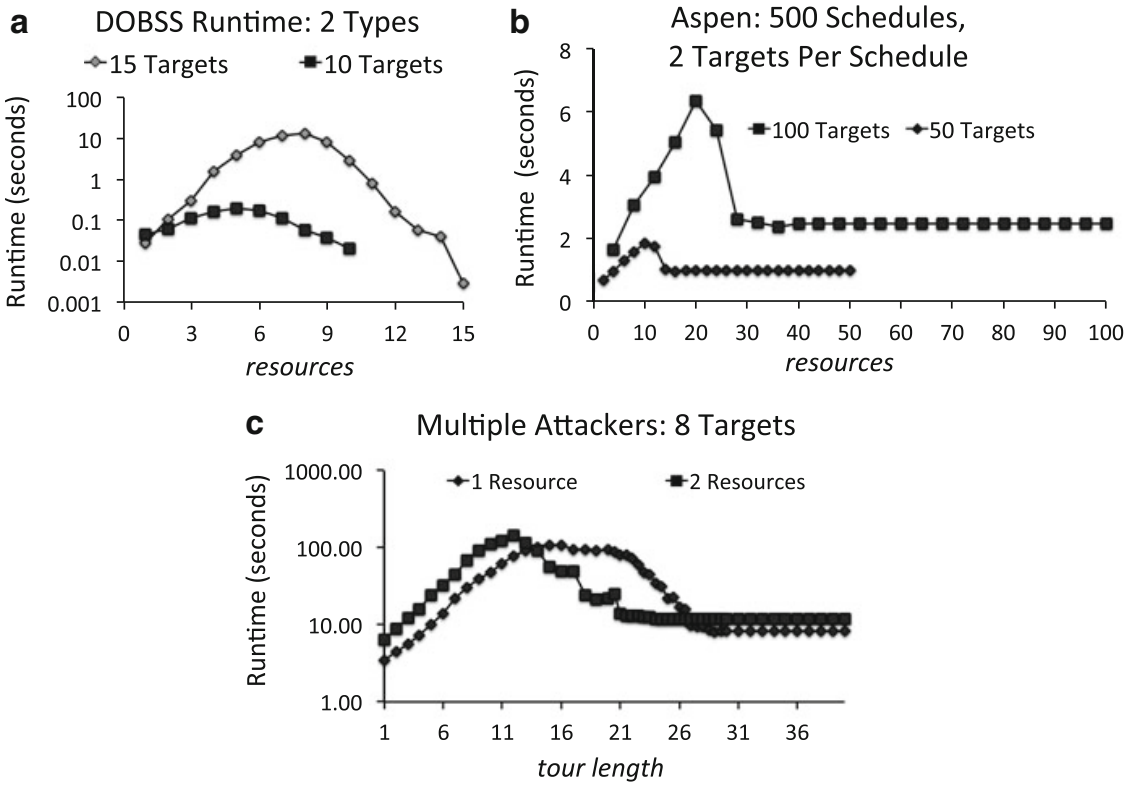

Fig. 2.7 Average running time of DOBSS, ASPEN and multiple-attacker branch-and-price algorithm for SPNSC, SPARS and SPPC domains respectively

(introduced in literature for these three domains [41]), five different underlying MIP solvers, two different equilibrium concepts in Stackelberg security games (ε-SSE [42] being the second equilibrium concept), and a variety of domain sizes and conditions.

More specifically, the deployment-to-saturation (d:s) ratio is defined in terms of *defender resources*, a concept whose precise definition differs from one domain to another. Given this definition, *deployment* denotes the number of defender resources available to be allocated, and *saturation* denotes the minimum number of defender resources such that the addition of further resources beyond this point yields no increase in the defender's expected utility. As mentioned earlier, for the ARMOR and the IRIS domains, deployment denotes the number of available security personnel, whereas saturation refers to the minimum number of officers required to cover all the targets with a probability of 1. For the PROTECT domain, deployment denotes the maximum feasible tour length, while saturation denotes the minimum tour length required by the team of defender resources such that the team can tour all the targets with a probability of 1. Sample results for these three domains are shown in Fig. 2.8, where the x-axis is the d:s ratio and the primary y-axis is runtime in seconds. The secondary y-axis and the dashed lines show a *phase transition* that we describe later. The important deduction to draw from these graphs is that the hardest computation is required when the d:s ratio is close to 0.5.

There are two important implications of this finding. First, new algorithms should be compared on the hardest problem instances; indeed, most previous

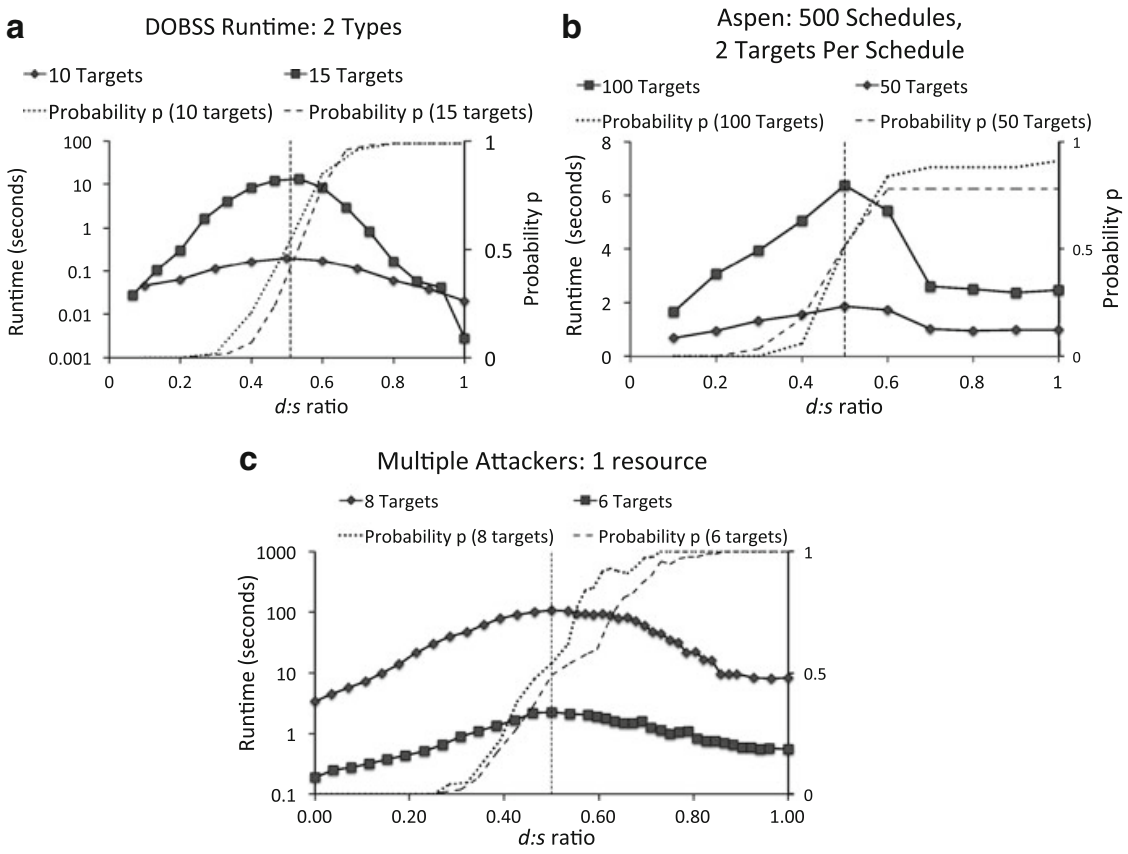

Fig. 2.8 Average runtime for computing the optimal solution for an example setting for all the three security domains, along with the probability p plotted on the secondary y-axis. The vertical dotted line shows $d{:}s = 0.5$

research has compared the runtime performance of algorithms only at low $d{:}s$ ratios, where problems are comparatively easy [41]. Second, this computationally hard region is the point where optimization offers the greatest benefit to security agencies, implying that problems in this region deserve increased attention from researchers [41]. Furthermore, we present an analysis of the runtime results using the concept of a *phase transition* in the decision version of SSE optimization problem.

All the runtime results show an easy-hard-easy computational pattern as the $d{:}s$ ratio increases from 0 to 1, with the hardest problems at $d{:}s = 0.5$. Such easy–hard–easy patterns have also been observed in other NP-complete problems, most notably 3-SAT [43, 44]. In 3-SAT, the hardness of the problems varies with the clause-to-variable (c/v) ratio, with the hardest instances occurring at about $c/v = 4.26$. The SAT community has used the concept of phase transitions to better understand this hardness peak. Phase transitions are defined for decision problems; and the decision version $SSE(D)$ of the SSE optimization problem asks whether there exists a defender strategy that guarantees expected utility of at least the given value D.

The results plotting the phase transition are also shown in Fig. 2.8. The x-axis shows the $d{:}s$ ratio, the primary y-axis shows the runtime in seconds, and the

secondary y-axis shows the probability p of finding a solution to $SSE(D^*)$. D^* was chosen by computing the median defender utility for 100 random problem instances for the domain [41]. The runtimes are plotted using solid lines, and p is plotted using a dashed line. Figure 2.8a presents results for the DOBSS algorithm [13] for the ARMOR domain for 10 targets and 2 and 3 attacker types. As expected, the d:s ratio of 0.5 corresponds with $p = 0.51$ as well as the computationally hardest instances; more interestingly, p undergoes a phase transition as the d:s grows. Similarly, Fig. 2.8b shows results for the ASPEN algorithm [23] for the IRIS domain with 500 schedules, 2 targets per schedule and 50 and 100 schedules, and Fig. 2.8c shows results for the multiple attacker PROTECT domain [41] for 8 targets. In both cases, we again observe a phase transition in p.

2.5 Open Research Issues

While the deployed applications have advanced the state of the art, significant future research remains to be done. In the following, we highlight some key research challenges, including scalability, robustness, human adversary modeling, and mixed-initiative optimization. The main point we want to make is that this research does not require access to classified information of any kind. Problems, solution approaches and datasets are well specified in the papers discussed below,

Scalability: The first research challenge is improving the scalability of our algorithms for solving Stackelberg (security) games. The strategy space of both the defender and the attacker in these games may exponentially increase with the number of security activities, attacks, and resources. As we scale up to larger domains, it is critical to develop newer algorithms that scale up significantly beyond the limits of the current state of the art of Bayesian Stackelberg solvers. Driven by the growing complexity of applications, a sequence of algorithms for solving security games have been developed including DOBSS [13], ERASER [16], ASPEN [23], HBGS [17] and RUGGED [36]. However, existing algorithms still cannot scale up to very large scale domains such as scheduling randomized checkpoints in cities (while RUGGED computes optimal solutions much faster than any of the previous approaches, much work remains to be done for it to be applicable on a large urban road network).

Robustness: The second challenge is improving solutions' robustness. Classical game theory solution concepts often make assumptions on the knowledge, rationality, and capability (e.g., perfect recall) of players. Unfortunately, these assumptions could be wrong in real-world scenarios. Therefore, while computing the defender's optimal strategy, algorithms should take into account various uncertainties faced in the domain, including payoff noise [45], execution/observation error [46], uncertain capability [47]. While there are algorithms for dealing with different types of uncertainties, there is no general algorithm/framework that can deal with different types of uncertainty simultaneously. Furthermore, existing work assumes that the attacker knows (or with a small noise) the defender's strategy and

there is no formal framework to model the attacker's belief update process and how it makes tradeoffs in consideration of surveillance cost, which remains an open issue for in future research.

One required research direction with respect to robustness is addressing bounded rationality of human adversaries, which is a fundamental problem that can affect the performance of our game theoretic solutions. Recently, there has been some research on applying ideas (e.g., prospect theory [48], and quantal response [24]) from social science or behavioral game theory within security game algorithms [42, 49]. Previous work usually applies existing frameworks and sets the parameters of these frameworks by experimental tuning or learning. However, in real-world security domains, we may have very limited data, or may only have some limited information on the biases displayed by adversaries. It is thus still a challenging problem to build high fidelity human attacker models that can address human bounded rationality. Furthermore, since real-world human attackers are sometimes distributed coalitions of socially, culturally and cognitively-biased agents, acting behind a veil of uncertainty, we may need significant interdisciplinary research to build in social, cultural and coalitional biases into our adversary models.

Mixed-Initiative Optimization: Another challenging research problem in security games is mixed-initiative optimization in which human users and software assistants collaborate to make security decisions [50]. There often exist different types of constraints in security applications. For instance, the defender always has resource constraints, e.g., the numbers of available vehicle checkpoints, canine units, or air marshals. In addition, human users may place constraints on the defender's actions to affect the output of the game when they are faced with exceptional circumstances and extra knowledge. For instance, in the ARMOR system there could be forced checkpoints (e.g., when the Governor is flying) and forbidden checkpoints. Existing applications simply compute the optimal solution to meet all the constraints (if possible). Unfortunately, these user defined constraints may lead to poor (or infeasible) solutions due to the users' bounded rationality and insufficient information about how constraints affect the solution quality. Significantly better solution quality can be obtained if some of these constraints can be relaxed. However, there may be infinitely many ways of relaxing constraints and the software assistant may not know which constraints can be relaxed and by how much, as well as the real-world consequences of relaxing some constraints.

Thus, it is promising to adopt a mixed-initiative approach in which human users and software assistants collaborate to make security decisions. However, designing an efficient mixed-initiative optimization approach is not trivial and there are five major challenges. First, the scale of security games and constraints prevent us from using an exhaustive search algorithm to explore all constraint sets. Second, the user's incomplete information regarding the consequences of relaxing constraints requires preference elicitation support. Third, the decision making of shifting control between the user and the software assistant is challenging. Fourth, it is difficult to evaluate the performance of a mixed-initiative approach. Finally, it is a challenging problem to design good user interfaces for the software assistant to

explain how constraints affect the solution quality. What remains to be done for the mixed-initiative approach includes sensitivity analysis for understanding how different constraints affect the solution quality, inference/learning for discovering directions of relaxing constraints, search for finding constraint sets to explore, preference elicitation for finding the human user's preference of different constraint sets, and interface design for explaining the game theoretic solver's performance.

Multi-objective Optimization: In existing applications such as ARMOR, IRIS and PROTECT, the defender is trying to maximize a single objective. However, there are domains where the defender has to consider multiple objectives simultaneously. For example, the Los Angeles Sheriff's Department (LASD) needs to protect the city's metro system from ticketless travelers, common criminals, and terrorists. From the perspective of LASD, each one of these attacker types provides a unique threat (lost revenue, property theft, and loss of life). Given this diverse set of threats, selecting a security strategy is a significant challenge as no single strategy can minimize the threat for all attacker types. Thus, tradeoffs must be made and protecting more against one threat may increase the vulnerability to another threat. However, it is not clear how LASD should weigh these threats when determining the security strategy to use. One could attempt to establish methods for converting the different threats into a single metric. However, this process can become convoluted when attempting to compare abstract notions such as safety and security with concrete concepts such as ticket revenue.

Multi-objective security games (MOSG) have been proposed to address the challenges of domains with multiple incomparable objectives [51]. In an MOSG, the threats posed by the attacker types are treated as different objective functions which are not aggregated, thus eliminating the need for a probability distribution over attacker types. Unlike Bayesian security games which have a single optimal solution, MOSGs have a set of pareto-optimal (non-dominated) solutions which is referred to as the Pareto frontier. By presenting the pareto frontier to the end user, they are able to better understand the structure of their problem as well as the trade-offs between different security strategies. As a result, end users are able to make a more informed decision on which strategy to enact. Existing approaches so far assume that each attacker type has a single objective and there is no uncertainty regarding each attacker type's payoffs. It is challenging to develop algorithms for solving multi-objective security games with multiple attacker objectives and uncertain attacker payoffs.

In addition to the above research challenges, there are other on-going challenges such as preference elicitation for acquiring necessary domain knowledge in order to build game models and evaluation of the game theoretic applications [52].

Acknowledgements This research is supported by MURI grant W911NF-11-1-0332, ONR grant N00014-08-1-0733 and by the United States Department of Homeland Security through the Center for Risk and Economic Analysis of Terrorism Events (CREATE) under grant number 2010-ST-061-RE0001. All opinions, findings, conclusions and recommendations in this document are those of the authors and do not necessarily reflect views of the United States Department of Homeland Security.

References

1. Pita, J., Jain, M., Western, C., Portway, C., Tambe, M., Ordonez, F., Kraus, S., Parachuri, P.: Deployed ARMOR protection: The Application of a Game-Theoretic Model for Security at the Los Angeles International Airport. In: Proc. of The 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2008) 125–132
2. Tsai, J., Rathi, S., Kiekintveld, C., Ordonez, F., Tambe, M.: IRIS: a Tool for Strategic Security Allocation in Transportation Networks. In: Proc. of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2009) 37–44
3. An, B., Pita, J., Shieh, E., Tambe, M., Kiekintveld, C., Marecki, J.: GUARDS and PROTECT: Next Generation Applications of Security Games. SIGECOM **10** (March 2011) 31–34
4. 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: Proc. of The 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2012)
5. Pita, J., Tambe, M., Kiekintveld, C., Cullen, S., Steigerwald, E.: GUARDS - Game Theoretic Security Allocation on a National Scale. In: Proc. of The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2011)
6. Yin, Z., Jiang, A., Johnson, M., Tambe, M., Kiekintveld, C., Leyton-Brown, K., Sandholm, T., Sullivan, J.: TRUSTS: Scheduling Randomized Patrols for Fare Inspection in Transit Systems. In: Proc. of The 24th Conference on Innovative Applications of Artificial Intelligence (IAAI). (2012)
7. von Stackelberg, H.: Marktform und Gleichgewicht. Springer, Vienna (1934)
8. Gatti, N.: Game Theoretical Insights in Strategic Patrolling: Model and Algorithm in Normal-Form. In: ECAI-08. (2008) 403–407
9. Lye, K., Wing, J.M.: Game Strategies in Network Security. International Journal of Information Security **4**(1–2) (2005) 71–86
10. Brown, G., Carlyle, M., Kline, J., Wood, K.: A Two-Sided Optimization for Theater Ballistic Missile Defense. In: Operations Research. Volume 53. (2005) 263–275
11. Sandler, T., M., D.G.A.: Terrorism and Game Theory. Simulation and Gaming **34**(3) (2003) 319–337
12. Avenhaus, R., von Stengel, B., Zamir, S.: Inspection Games. In Aumann, R.J., Hart, S., eds.: Handbook of Game Theory. Volume 3. North-Holland, Amsterdam (2002) 1947–1987
13. Paruchuri, P., Pearce, J.P., Marecki, J., Tambe, M., Ordonez, F., Kraus, S.: Playing Games with Security: An Efficient Exact Algorithm for Bayesian Stackelberg Games. In: Proc. of The 7th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2008) 895–902
14. Conitzer, V., Sandholm, T.: Computing the Optimal Strategy to Commit to. In: Proc. of the ACM Conference on Electronic Commerce (ACM-EC). (2006) 82–90
15. Brown, G., Carlyle, M., Salmeron, J., Wood, K.: Defending Critical Infrastructure. In: Interfaces. Volume 36. (2006) 530 – 544
16. Kiekintveld, C., Jain, M., Tsai, J., Pita, J., Tambe, M., Ordonez, F.: Computing Optimal Randomized Resource Allocations for Massive Security Games. In: Proc. of The 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2009) 689–696
17. Jain, M., Kiekintveld, C., Tambe, M.: Quality-Bounded Solutions for Finite Bayesian Stackelberg Games: Scaling Up. In: Proc. of The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2011)
18. Leitmann, G.: On Generalized Stackelberg Strategies. Optimization Theory and Applications **26**(4) (1978) 637–643
19. Breton, M., Alg, A., Haurie, A.: Sequential stackelberg equilibria in two-person games. Optimization Theory and Applications **59**(1) (1988) 71–97
20. von Stengel, B., Zamir, S.: Leadership with Commitment to Mixed Strategies. Technical Report LSE-CDAM-2004-01, CDAM Research Report (2004)

21. Osbourne, M.J., Rubinstein, A.: A Course in Game Theory. MIT Press (1994)
22. Jain, M., Tsai, J., Pita, J., Kiekintveld, C., Rathi, S., Tambe, M., Ordonez, F.: Software Assistants for Randomized Patrol Planning for the LAX Airport Police and the Federal Air Marshal Service. Interfaces **40** (2010) 267–290
23. Jain, M., Kardes, E., Kiekintveld, C., Ordonez, F., Tambe, M.: Security Games with Arbitrary Schedules: A Branch and Price Approach. In: Proc. of The 24th AAAI Conference on Artificial Intelligence. (2010) 792–797
24. McKelvey, R.D., Palfrey, T.R.: Quantal Response Equilibria for Normal Form Games. Games and Economic Behavior **10**(1) (1995) 6–38
25. TSA: Layers of Security: What We Do. http://www.tsa.gov/what_we_do/layers/index.shtm (2011)
26. TSA: Transportation Security Administration — U.S. Department of Homeland Security. http://www.tsa.gov/ (2011)
27. Hamilton, B.A.: Faregating Analysis. Report Commissioned by the LA Metro. http://boardarchives.metro.net/Items/2007/11_November/20071115EMACItem27.pdf (2007)
28. Jiang, A.X., Yin, Z., Kietkintveld, C., Leyton-Brown, K., Sandholm, T., Tambe, M.: Towards Optimal Patrol Strategies for Urban Security in Transit Systems. In: Proc. of the AAAI Spring Symposium on Game Theory for Security, Sustainability and Health. (2012)
29. Chandran, R., Beitchman, G.: Battle for Mumbai Ends, Death Toll Rises to 195. Times of India (29 November 2008) http://articles.timesofindia.indiatimes.com/2008-11-29/india/27930171_1_taj-hotel-three-terrorists-nariman-house.
30. Johnson, M., Fang, F., Yang, R., Tambe, M., Albers, H.: Patrolling to Maximize Pristine Forest Area. In: Proc. of the AAAI Spring Symposium on Game Theory for Security, Sustainability and Health. (2012)
31. Ordonez, F., Tambe, M., Jara, J.F., Jain, M., Kiekintveld, C., Tsai, J.: Deployed Security Games for Patrol Planning. In: Handbook on Operations Research for Homeland Security. (2008)
32. Trusov, M., Bucklin, R.E., Pauwels, K.: Effects of Word-of-Mouth versus Traditional Marketing: Findings from an Internet Social Networking Site. Journal of Marketing **73** (2009)
33. Howard, N.J.: Finding Optimal Strategies for Influencing Social Networks in Two Player Games. Master's thesis, MIT, Sloan School of Management (2011)
34. Alpcan, T.: Network Security: A Decision and Game-Theoretic Approach. Cambridge University Press (2010)
35. Vanek, O., Yin, Z., Jain, M., Bosansky, B., Tambe, M., Pechoucek, M.: Game-Theoretic Resource Allocation for Malicious Packet Detection in Computer Networks. In: Proc. of The 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2012)
36. Jain, M., Korzhyk, D., Vanek, O., Pechoucek, M., Conitzer, V., Tambe, M.: A Double Oracle Algorithm for Zero-Sum SEcurity games on Graphs. In: Proc. of The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2011)
37. Keteyian, A.: TSA: Federal Air Marshals. (2010) http://www.cbsnews.com/stories/2010/02/01/earlyshow/main6162291.shtml, *retrieved* Feb 1, 2011.
38. Bertsimas, D., Tsitsiklis, J.N.: Introduction to Linear Optimization. Athena Scientific (1994)
39. Yin, Z., Korzhyk, D., Kiekintveld, C., Conitzer, V., Tambe, M.: Stackelberg vs. Nash in security games: interchangeability, equivalence, and uniqueness. In: AAMAS. (2010) 1139–1146
40. Letchford, J., Conitzer, V., Munagala, K.: Learning and Approximating the Optimal Strategy to Commit To. In: Second International Symposium on Algorithmic Game Theory (SAGT). (2009) 250–262
41. Jain, M., Leyton-Brown, K., Tambe, M.: The Deployment-to-Saturation Ratio in Security Games. In: Proc. of The 26th AAAI Conference on Artificial Intelligence (AAAI). (2012)
42. Pita, J., Jain, M., Tambe, M., Ordóñez, F., Kraus, S.: Robust Solutions to Stackelberg Games: Addressing Bounded Rationality and Limited Observations in Human Cognition. Artificial Intelligence **174**(15) (2010) 1142–1171
43. Cheeseman, P., Kanefsky, B., Taylor, W.M.: Where the Really Hard Problems are. In: Proceedings of the International Joint Conference on Artificial Intelligence. (1991) 331–337

44. Mitchell, D., Selman, B., Levesque, H.: Hard and Easy Distributions of SAT Problems. In: Proceedings of the American Association for Artificial Intelligence. (1992) 459–465
45. Kiekintveld, C., Marecki, J., Tambe, M.: Approximation Methods for Infinite Bayesian Stackelberg Games: Modeling Distributional Uncertainty. In: Proc. of The 10th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2011)
46. Yin, Z., Jain, M., Tambe, M., Ordonez, F.: Risk-Averse Strategies for Security Games with Execution and Observational Uncertainty. In: Proc. of The 25th AAAI Conference on Artificial Intelligence (AAAI). (2011) 758–763
47. An, B., Tambe, M., Ordonez, F., Shieh, E., Kiekintveld, C.: Refinement of Strong Stackelberg Equilibria in Security Games. In: Proc. of the 25th Conference on Artificial Intelligence. (2011) 587–593
48. Kahneman, D., Tvesky, A.: Prospect Theory: An Analysis of Decision Under Risk. Econometrica **47**(2) (1979) 263–291
49. Yang, R., Kiekintveld, C., Ordonez, F., Tambe, M., John, R.: Improving Resource Allocation Strategy Against Human Adversaries in Security Games. In: IJCAI. (2011)
50. An, B., Jain, M., Tambe, M., Kiekintveld, C.: Mixed-Initiative Optimization in Security Games: A Preliminary Report. In: Proc. of the AAAI Spring Symposium on Help Me Help You: Bridging the Gaps in Human-Agent Collaboration. (2011) 8–11
51. Brown, M., An, B., Kiekintveld, C., Ordonez, F., Tambe, M.: Multi-objective optimization for security games. In: Proc. of The 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS). (2012)
52. Taylor, M.E., Kiekintveld, C., Western, C., Tambe, M.: A Framework For Evaluating Deployed Security Systems: Is There a Chink in your ARMOR? Informatica **34** (2010) 129–139

http://www.springer.com/978-1-4614-5415-1

Moving Target Defense II
Application of Game Theory and Adversarial Modeling
Jajodia, S.; Ghosh, A.K.; Subrahmanian, V.S.; Swarup, V.;
Wang, C.; Wang, X.S. (Eds.)
2013, XII, 204 p., Hardcover
ISBN: 978-1-4614-5415-1