Evaluating Deployed Decision Support Systems for Security: Challenges, Analysis, and Approaches

Matthew E. Taylor, Christopher Kiekintveld, and Milind Tambe
Lafayette College, Computer Science Department, *taylorm@lafayette.edu*
The University of Texas at El Paso, Computer Science Department, *cdkiekintveld@utep.edu*
The University of Southern California, Computer Science Department, *tambe@usc.edu*

Abstract

This chapter discusses the difficult problem of evaluating deployed security-focused decision support systems. In a security setting, one can never expect the adversary to cooperate in evaluation, which disallows many kinds of controlled studies. Furthermore, data is typically sparse — there are thankfully relatively few terrorist attacks on major infrastructure in the U.S. Still, evaluating security measures is critical in ensuring efficient allocation of security resources. We discuss a variety of approaches for evaluating such systems, using the deployed ARMOR and IRIS systems as exemplars. Taken as a whole, the evidence supports the effectiveness of these systems, but there are clearly opportunities to improve our methods for assessing the value proposition of all types of security systems.

1 Introduction

As discussed in other chapters of this book, there are an increasing number of technically sophisticated tools to support decision-making for security resource allocation in many different domains. In this chapter we discuss the question of evaluating these deployed security systems, using examples from our own research to illustrate some of the key challenges in doing evaluation for security systems. Two of the primary difficulties are that (1) we cannot rely on adversaries to cooperate in evaluation, which makes it difficult to validate models, and (2) there is (thankfully) very little data available about real-world terrorist attacks.

Despite the difficulties of comprehensive evaluation in security domains, it is only through asking the question "how well does a system work?" that policy makers can decide how to allocate finite resources to to different security measures. In this chapter we discuss the goals of security systems, the elements that comprise these systems, and different approaches for evaluation. Every approach has drawbacks, so in lieu of an ideal test we advocate a comprehensive style of evaluation that uses diverse metrics and data to perform cost-benefit analysis for the complete system. We also emphasize that the focus of the evaluation is not "is system X the perfect security system?" which is an impossible standard. Rather, the relevant question is which of the available alternatives should be used; providing strong evidence that one alternative is superior to other approaches is often feasible, even when providing exact quantitative measures of value is not. As a community, we must strive to perform the best possible evaluations using the tools and data available, but we cannot let the absence of an ideal evaluation prevent us from deploying effective technologies. However, we must also recognize weaknesses in the current evaluations, and view these as opportunities to develop new techniques and gather new data to improve our understanding of the value of different approaches for security.

Section 2 will provide a brief background on the security applications discussed in this chapter in the context of evaluation — more detailed descriptions of these systems are provided in other chapters of the book. Section 3 describes the three steps involved in formulating a decision support system for security: abstracting the model, solving the model, and then accurately deploying the solution. Section 4 describes the different types of evaluations that have been conducted on two systems in our case study. Section 5 will discuss the goals of a deployed security system and the inherent difficulties in measuring the performance of such a system. For instance, unlike many types of technical applications, a security system does not have binary behavior; no security system is able to provide 100% protection and it does not make sense to say that it "does" or "does not" work. Instead, systems must be evaluated on basis of *risk reduction*, often through indirect measures such as increasing adversary cost and uncertainty, or reducing the effectiveness of an adversaries' attack.

Section 6 discusses the pros and cons of different evaluation techniques, tying together the discussion in Sections 3–5. Related work is discussed in Section 7. Finally, Section 8 ends the chapter with a discussion of future work, both in terms of enhancing the evaluation of our specific applications, as well as the challenge of security evaluation in general.

2 Background: ARMOR and IRIS

The importance of security games as a research topic is driven by the recent development and deployment of several applications that assist major security organizations with making resource allocation decisions using algorithmic analysis based on game theory. These include the ARMOR system deployed at the LAX International Airport (LAX) [Pita *et al.*, 2008] to assist the Los Angeles World Airport policy (LAWA) and the IRIS system deployed by the Federal Air Marshals Service (FAMS) [Tsai *et al.*, 2009]. While these systems share a similar approach in that they apply game-theoretic modeling techniques and emphasize randomized, unpredictable scheduling of security resources and activities, each system poses unique challenges in system design, modeling and solution techniques, and evaluation. ARMOR was the first system developed using this methodology, and has the longest deployment history — our discussion of evaluation in this chapter will focus largely on ARMOR with additional discussion of more recent systems where appropriate.

The ARMOR system has been deployed since 2007 by the Los Angeles World Airports (LAWA) police at LAX, the fifth busiest airport in the United States (and largest destination), serving 70–80 million passengers per year. LAX is considered a primary terrorist target on the West Coast and multiple individuals have been arrested for plotting or attempting to attack LAX [Stevens *et al.*, 2009]. Police have designed multiple "rings" of protection for LAX, including vehicular checkpoints, police patrols of roads, and inside terminals (some with bomb-sniffing canine units, also known as K9 units), passenger screening, and baggage screening. Due to the high demands on LAX security, due in part to the large physical area and high density of passenger and cargo traffic, police do not have enough resources (e.g., officers and K9 units) to provide constant security for every area and event at the airport. This limitation leads to the question of how best to allocate the limited resources to improve security at the airport.

ARMOR addresses two specific security problems by increasing the unpredictability of security schedules and weighting defensive strategy based on targets' importance. First, there are many roads that are entry points to LAX. When and where should vehicle checkpoints be set up on these roads? Pertinent information includes typical traffic patterns on inbound roads, the areas each road accesses within LAX, and areas of LAX which may have more or less importance as terrorist targets. Second, how and when should the K9 units patrol the eight terminals at LAX? Here it is important to consider the time-dependent passenger volumes per terminal, as well as the attractiveness of different terminals. In both cases, a predictable pattern can be exploited by an observant attacker.

The approach taken by ARMOR uses game-theoretic models to derive scheduling strategies for the police resources. This is modeled as a Bayesian Stackelberg game [Conitzer and Sandholm, 2006], in which the police (e.g., defenders) must commit to a (randomized) security policy. Each possible attacker type observes the security policy and then selects and optimal attack strategy based on their preferences (i.e., *utilities* or utility *payoff matrices*). The solution to this game is called a Strong Stackelberg Equilibrium, and yields an optimal randomized strategy for the policy. ARMOR uses an algorithm called DOBSS [Paruchuri *et al.*, 2008] to solve these game instances and recommend schedules for checkpoints and canine patrols to the LAWA police. The schedules account for three key factors: (1) attackers are able to observe the security policy using surveillance, (2) attackers change their behavior in response to the security policy, and (3) the risk/consequence of an attack varies depending on the target.

The IRIS system was designed to address scheduling problems faced by the Federal Air Marshals Service (FAMS). FAMS is charged with law enforcement and anti-terrorism on commercial airline flights, which is accomplished primarily by placing armed marshals on individual flights. There are tens of thousands of flights operated by US airlines each day, and the FAMS lack the resources to place marshals on every flight. This leads to a similar resource allocation challenge to that addressed by ARMOR: how best to place the available marshals on the possible flights to maximize security. In addition to the overall constraint on the number of air marshals available, this problem is complicated by the presence of complex physical and temporal constraints on which flight tours the air marshals can feasibly fly, as well as other idiosyncratic constraints. Flights are also diverse in the potential risk and consequence of a terrorist attack against the flight, based on many factors including the source and destination airports, aircraft size, and flight path. The schedules produced by IRIS must optimize over all of these constraints and variations in flight valuation, resulting in a significantly more complex problem space than for the ARMOR scheduler [Kiekintveld *et al.*, 2009]

3 Formulating the Problem

Having introduced ARMOR and IRIS, we now describe the basic elements of these decision-support tools, and the process used to design them. In each case, the complete system includes a domain model based on expert elicitation, a game-theoretic solution algorithm used to analyze the model, and a software application designed around the algorithm that allows data inputs, visualization of the solution, and so on. To fully evaluate the system we must evaluate each of these components, as well as the way the system is actually used in practice (e.g., how the recommended solutions are implemented by the security forces).

We focus first on the process of formulating a model of the domain, since this is the basis for all subsequent analysis. This model must have sufficient detail that it can provide useful decision support, it must accurately represent the domain, and it must not be so complex that it is intractable for analysis. The problem of computing a solution is discussed in the next section and is the most familiar to computer scientists. While other chapters have discussed the computational difficulties in solving complex security problems, our discussion considers more broadly the assumptions behind the solution methods. A system may be theoretically sound, but badly flawed if it is implemented poorly. The third and final step we discuss is implementing (and verifying) the system in a real-world setting.

3.1 Abstracting the Real World Problem

The first step of developing a defensive measure is to determine what particular attack vector should be addressed. For example, the ARMOR system was designed to counteract a perceived defensive shortcoming at LAX, as identified in a Rand study [Stevens *et al.*, 2009]. In order to construct a model that can be solved, the real problem must be formulated as a quantitative problem. In the case of ARMOR, this involved focusing on a fixed number of checkpoint positions that could be either covered or not covered. How many officers were at a checkpoint, or the skills of the particular officers is abstracted away.

In order to decide how important the different checkpoints were, they were assigned values based on their proximity and road layout relative to the terminals. The terminals, in turn, were evaluated by experts who estimated the utilities to the defenders if an attacker successfully attacked a terminal, or was caught attempting to attack a particular terminal. The experts also estimated the utilities to the attackers if they successfully attacked a terminal, or were caught attacking a particular terminal.

Formulating a real world problem as a solvable quantitative model requires abstraction, which by definition causes a loss of information. However, the goal is to minimize the loss of precision so that the optimal solution to the model will be near-optimal on the physical problem. The formulation chosen will also constrain the policy space. In ARMOR, checkpoints are constrained to last at least a certain amount of time, due to the time of setting up a checkpoint. Thus ARMOR could not schedule checkpoint #1 to be set up at 9:00 am and taken down at 9:05 am. The selection of this constrained solution space may also impact the optimality of the solution in the real world.

In addition to modeling the utilities for attacks, one often wishes to attempt to model deterrence. One method is to test an attacker's optimal option with and without defense. If the attacker changes actions to a lower-valued action, s/he can be considered to be deterred. Another option is to consider that the attacker may choose to not attack at all, or attack a target that is outside the scope of the security system in question. In a game-theoretic setting, one option is to add a "stay at home" action to the attacker's action space. A second option would be to add an "attack another target" option. In the case of ARMOR, this could include the action "attack another target in Los Angeles," or "attack another airport in the U.S." Notice that from the point of LAWA, these three possible actions are equivalent as they are all instances of successful deterrence with respect to LAX. However, for the city of Los Angeles or for the U.S. federal government, the "stay at home" action is clearly superior.

3.2 Solution Concepts and Computational Considerations

The previous section focused on the challenge of generating an abstract model of real-world problem that accurately represents the underlying situation. Once this model is generated, the next stage is to analyze this model to determine the best course of action (strategy) for the security forces. Given the complexity of the models we need fast computational methods to perform this analysis and generate recommendations.

Potential Solution Concepts

Given a precise mathematical model of a strategic interaction, the game theory literature provides many powerful tools for analyzing the model to make predictions and strategy recommendations. Unfortunately, there are many different solution concepts that may be applied in different situations, and it is not obvious which is the "best" solution method to use. The starting point is typically a form of Nash equilibrium, which predicts how player should act given that they are perfectly rational, and all players have exact common knowledge of the game model (including the possible actions, sequence of moves, payoffs for different outcomes, etc.). In real-world situations, these assumptions are often too strong, since players may be uncertain about different aspects of the game, or may not make mathematically optimal choices due to limits on their reasoning capabilities or other factors. Different solution concepts can be applied based on different assumptions about knowledge and rationality, but it is an important open question how well the solution concept and underlying assumptions correspond to the real world situation, and this must be evaluated empirically.

There are alternatives to game theoretic equilibrium solutions methods that should also be evaluated as candidates. The simplest is the uniform random strategy: given a set of possible security actions, take each of them with equal probability. This has the appeal of simplicity, and being completely unpredictable (by definition). It also make absolutely no assumptions about the adversary, so there is no chance that these assumptions may be incorrect or exploitable. However, it does not take into account the value of different actions, and may waste limited resources by performing less valuable actions too often. A weighted randomization can potentially account for this by selecting more valuable actions more frequently. The key question here is how to determine the weights for the randomization; in effect, a game-theoretic solution is one answer to how to find these weights in a principled way, since a game-theoretic solution is an instance of weighted randomization. There may be simpler approaches to finding a weighted randomization in some cases that could also be candidate strategies. Our evaluations of ARMOR and similar systems, discussed in subsequent sections, have consistently shown that game-theoretic solutions are superior to both uniform randomization and simple weighted randomization strategies.

The final point to mention regarding solution concepts is whether or not the attackers are assumed to be fully rational. For instance, in classical game theory settings, human behavior can be fully predicted if the utilities of the actors are known. In practice, however, humans are often not true "homo-economous" actors, but may make suboptimal decisions based on non-infinite reasoning abilities.

ARMOR's game-theoretic model uses strong assumptions about the attacker's rationality to predict how they will behave and optimize accordingly. Humans often do not always conform to the predictions of strict equilibrium models (though some other models offer better predictions of behavior [Erev *et al.*, 2002]). In addition, ARMOR assumes that an attacker can perfectly observe the security policy, which may not be possible in reality.

Algorithmic Goals

Computer science often focuses on developing highly efficient algorithms with low worst-case complexity (or proving that such efficient algorithm are not possible for a particular problem). For instance, a computer scientist may ask "can a solution be found in time polynomial in the number of joint actions" or "can a solution be found with an amount of memory polynomial in the number of joint actions?" Different methods for solving games can be compared against each other by measuring their running time, their memory usage, their scalability, etc. However, when discussing real world problems, a more pertinent question is "can a solution be found given our constraints?"

In the ARMOR system, schedules for checkpoints are generated once a week on a standard desktop computer. This means that, given the ARMOR problem definition, the solution method must be able to finish within one week with a relatively pedestrian processor and less than 4GB of memory available. Thus, speeding up a solution method from nine days to three days is much more important than working to speed up the system from a run time of three days down to a single day.[1] Thus the standard metrics for evaluating algorithms may be less important than the practical requirements of a given model. Of particular note is that the worst-case complexity of the algorithms is largely irrelevant; the practical performance on typical instance is far more important.

Optimality proofs (or approximation guarantees) are also highly valuable in real-world applications. Without such guarantees it is difficult to build confidence in the system, and there is a legitimate concern that if an adverse event occurs it could be attributed to a poor-quality solution. While such proofs are often necessary, they are not sufficient to prove that a security system works. First, those in non-technical fields (who are necessarily part of the system's funding, implementation, and staffing), may not value the results of a proof as highly as a mathematician. Second, just because the model has been optimally solved does not mean that the correct problem was modeled (Section 3.1) or that the solution was correctly implemented (Section 3.3).

3.3 Implementing the Solution

The final element to consider is how the tool is actually used in practice. Any decision support system requires inputs from human users, and and the recommendations it makes must be implemented by the users as well. Problems at the interface between humans and the computer system may cause the overall system to be ineffective, even if the computer system is theoretically perfect! Based on our personal experience with implemented systems, we believe that a tight loop between the modelers, system developers, and end users is critical to a successful system. For instance, if those building the model do not understand how the solution will be implemented, they may not account for key factors in the real world problem. Likewise, if assumptions made when solving the model are violated, any optimality guarantees are invalidated.

For example, the ARMOR system schedules checkpoints at certain times. During our discussions with LAWA, we heard anecdotal evidence that before ARMOR, some checkpoints were typically not manned during the middle of the day because they were in direct sunlight and uncomfortably hot. This is precisely the kind of predictable behavior that an adversary could exploit, and which the ARMOR system is designed to avoid. However, ARMOR produced a schedule calling for one of these checkpoints to be manned during the middle of the day and the schedule was violated, the assumptions made by ARMOR would be invalidated. (To the best of our knowledge, this has not happened. In fact, we have heard that officers stationed to such checkpoints previously complained to the scheduling officer, but now complaints have been reduced because it was "the computer's decision," rather than the decision of a senior officer.)

While the above guideline is true of all deployed systems, security presents two unique challenges. First, those who perform the modeling and solving steps may not be able to make site visits and fully understand the proposed deployment. For instance, a game theory expert who is working on developing safe convoy routes for patrols in Iraq is unlikely to want to travel to Baghdad given the security risks. Further, in some cases, the full details of the defensive operation may not be available to the modelers or model solvers for security reasons, and they must work under assumptions that those with appropriate security clearances independently verify.

Second, data related to the deployed configuration and performance of the system may be classified or sensitive, and not available to researches. For instance, when working with the FAMS, we have developed a solution method that takes the utility matrices as input and outputs a schedule. However, as academic computer scientists, we do not have clearance to see the true utility matrices used by the FAMS, nor are we able to view the schedules produced by the production system. This enforced disconnect means that it is impossible for all members of the designing team to understand the full operation, and thus implementation and verification details fall most heavily on those security clearance to see the full picture.

Other questions related to implementation can best be answered through interviews and surveys of users. For instance:

- Do the users understand the day to day use of the system?
- Do users consistently enter correct inputs?
- Do users follow the recommendations of the system?
- What happens if there are unanticipated events (e.g., flight cancellations/delays, emergency situations, people call in sick, etc.)? Does the decision support system effectively handle such situations?

4 Evaluation Case Studies

Having introduced two examples of decision support systems for security and discussed the components of these systems, we now explore some of the existing evaluations we have performed on these systems. Latter sections of this chapter will refer back to these examples when discussing the overall effectiveness of the systems and delving into the strengths and weaknesses of different types of evaluations and metrics.

[1]Of course, faster run times make the system easier to work with, particularly in pre-deployment testing phases.

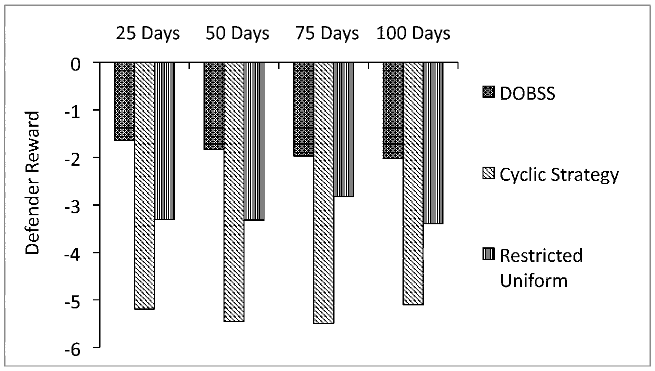


Figure 1: This figure compares ARMOR strategies (which use the DOBBS algorithm) and policies representative of previous methods where adversaries have the option to attack once every 25 days.

4.1 Comparison with Previous Best Practices

The most straightforward type of evaluation is to assume that the model is correct, and evaluate different potential strategies using the model. In particular, we can test the solutions used in ARMOR and IRIS against simpler randomization approaches, such as uniform random or simple heuristic weighted randomization. We can also test against strategies designed to replicate the best practices used by security forces. For example, by comparing the schedules generated by ARMOR with the previous scheduling practices at LAX [Cruz, 2009], we can answer the question "does ARMOR improve security" with more confidence.

There are some clear differences in ARMOR schedules and the previous scheduling practices at LAX. First, in the previous scheduling system, all checkpoints remained in place for an entire day, whereas checkpoints are now are moved throughout the day according to ARMOR's schedule (adding to the adversary's uncertainty). Second, before ARMOR only a single checkpoint was manned on any given day; multiple checkpoints are now used (due in part to an increased security budget). Third, a fixed sequence of checkpoints was defined (i.e., checkpoints 2, 3, 1, etc.), to create a static mapping from date to checkpoint. This sequence was not optimized according to the importance of different targets and the sequence would repeat (allowing the attacker to anticipate which checkpoint would be manned on any given day).

The LAX officers informed us that they previously generated checkpoint schedules based on a cyclic strategy with random perturbations. A study of past schedules showed that patterns remained evident despite these random perturbations — no checkpoint was repeated for two consecutive days. Therefore, we also compared the ARMOR strategy against two strategies: (1) a "cyclic" strategy where the checkpoints were scheduled in a cyclic order on all inbound roads and, (2) a "restricted uniform." strategy which was a uniformly random strategy with the additional restriction that no checkpoint was repeated on two consecutive days.

Our first experiments attempt to replicate as closely as possible the scheduling policies in use at LAX prior to ARMOR. Police officers place one checkpoint on any of the five inbound roads, and the rewards of the terminals were randomly chosen between 1 and 10. We vary the duration for which the adversary can make observations from 0 to 100 days, in increments of 25 days. We averaged our results over 100 trials. In these simulations, we assume that the adversary can use simple pattern recognition techniques to recognize patterns in the checkpoint schedules. The adversary maintains a history of observed checkpoints and generates confidence intervals over sequences of observations.

Figure 1 shows our experimental results. The x-axis represents the number of observations available to the adversary, and the y-axis represents the average defender reward. The ARMOR strategy has a higher average defender reward compared to the other two strategies. The reason is that the adversary can better predict the defender action in case of 'cyclic' and 'restricted uniform' strategies as compared to the ARMOR strategy. Therefore, simple pattern recognition techniques are sufficient for the adversary to exploit the patterns in the cyclic and restricted uniform strategies. These patterns can be avoided by the use of uniformly random strategies, but a uniform random strategy does not take into account the different preferences over targets of the defender. ARMOR provides weights for the different targets such that the average defender reward is the highest when compared against both cyclic and restricted uniform strategies. Additionally, ARMOR strategies are not just weighted random since they also account for the fact that the adversary observes the defender's strategy and then makes an informed rational choice.

Comparing schedules generated by ARMOR against a benchmark uniform random schedule shows that ARMOR's schedule is much more efficient. For example, Figure 2 shows the expected reward for the police using ARMOR's schedule (calculated using DOBSS) compared with a uniform random benchmark strategy in the canines domain. ARMOR is able

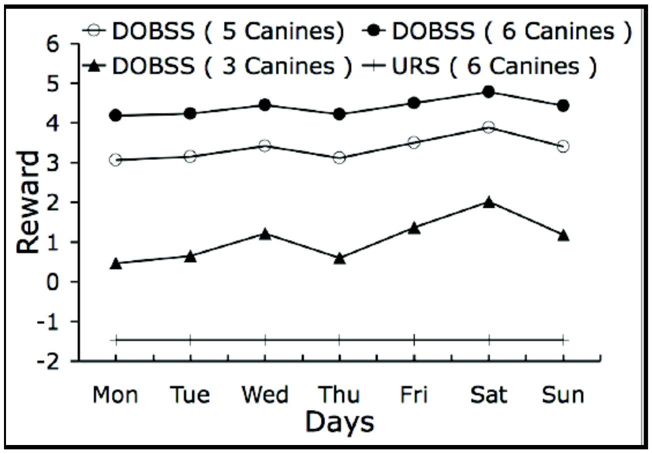


Figure 2: This figure shows that ARMOR (powered by the DOBBS algorithm) can outperform the baseline solution method, even using many fewer K9 units, which may achieve substantial cost savings. The x-axis shows the results from seven different days, and the y-axis shows the expected utility for the different scheduling methods.

to make such effective use of resources that using three canines scheduled with DOBSS yields higher utility than using six canines with uniform random scheduling!

Scheduling for the FAMS domain has undergone multiple mathematical tests using real-world schedules and withstood expert scrutiny, but the exact results of these tests are not released. Figure 3(a) shows the results of a mathematical evaluation done with hypothetical payoffs, to better measure the success of the system. Results show the expected payoff when scheduling a single air marshal in the FAMS domain over 20, 100, and 200 schedules. In these experiments, this equated to 10, 50, and 100 flights, with half as "departures" and half as "return" flights for the air marshals. The x-axis in these graphs show the expected payoff for IRIS, 6 different methods of weighting the schedules, uniform random, and no coverage (i.e., the single air marshal protects no planes).

In the full FAMS problem, there are too many schedules to enumerate and it is difficult to decide how to weight the flights, even if enumerated. This weighting is a simple way of determining which schedules to cover, and may improve the performance of the defender relative to uniform random. In these simple tests, the problem space was small enough that all schedules could be easily enumerated.

Utilities are generated as random numbers drawn uniformly from the ranges of numbers as follows:

- Defender payoff for covering a flight that is attacked: [+5000, +10000]
- Defender payoff for not covering a flight that is attacked: [-10000, -5000]
- Attacker payoff for attacking a flight that is not covered: [+5000, +10000]
- Attacker payoff for attacking a flight that is covered: [-10000, -5000]

The six naive weighted random strategies are as follows. In all cases a mixed probability distribution is obtained by normalizing these weights.

Max. of attacker reward The weight of a schedule is the maximum of the attacker rewards for a successful attack over all flights covered in the schedule.

Min. of attacker penalty The weight of a schedule is the minimum of the attacker penalty for a failed attack over all flights covered in the schedule.

Min. of defender penalty The weight of a schedule is the minimum of defender penalties for a successful attack over all flights covered in the schedule.

Max. of defender reward The weight of a schedule is the maximum of defender rewards for capturing an attacker over for all flights covered in the schedule.

Sum of defender penalties The weight of a schedule is the sum of the defender penalties for a successful attack over all flights covered in the schedule.

Sum of defender rewards The weight of a schedule is sum of defender rewards for capture over all flights covered in the schedule.

4.2 Mathematical Sensitivity Analysis

The analyses in the previous section compare the effectiveness of different scheduling policies, but do so based on all of the assumptions in the model (e.g., that the actions and payoffs are correctly specified for both players, and that the attacker is perfectly rational). To build more confidence in the approach, we must also validate the model itself. A first step in this is to better understand the impact of the different assumptions using sensitivity analysis.

In this type of evaluation, important parameters of the model are varied to test how the output of the model changes to different inputs. One important input to the ARMOR model is the distribution of different types of attackers. For example, some attackers may be highly organized and motivated, while others are amateurish and more likely to surrender. Different types of attackers can be modeled as having different payoff matrices. Changing the percentages of each attacker can help show the system's sensitivity to assumptions regarding the composition of likely attackers, and (indirectly) the system's dependence on precise utility elicitation. In Figure 4(a)–4(c), there are two adversary types with different reward matrices. Figure 4(a) demonstrates that DOBSS has a higher expected utility than that of a uniform random strategy on a single checkpoint, regardless of the percentage of "type one" and "type two" adversaries. Figures 4(b) and (c) shows that DOBSS again dominates uniform random for two and three checkpoints, respectively.

Further sensitivity analysis can be applied to other parameters of the model. The payoffs that describe the preferences of the two players for different outcomes are a very important set of parameters. These parameters are estimates of the true utilities determined through elicitation sessions with experts. Unfortunately, it is known that game-theoretic models can be quite sensitive to payoff noise [Kiekintveld and Wellman, 2008], and arbitrary changes in payoffs can lead to arbitrary changes in the optimal schedule. There is some evidence that ARMOR is robust to certain types of variations. In one experiment, we multiplied all of the defender's negative payoffs for successful attacks by a factor of four, essentially increasing the impact of a successful attack. We found that in the one and three checkpoint case, the strategies were unchanged. In the two checkpoint case the actions were slightly different, but the overall strategy and utility were unchanged. Unfortunately, there is also evidence that this does not generalize to all payoffs in security games. Kiekintveld et al. [Kiekintveld *et al.*, 2011] show that in general, adding small amounts of noise to the attacker's payoffs in security games can cause large deviations in the defender's payoffs (though the changes in the optimal strategy are less drastic).

4.3 Human Trials

Another set of assumptions in the game models use for ARMOR and IRIS is that the attackers are perfectly rational, and will always choose the mathematically optimal attack strategy. To test the sensitivity of the solutions to variations in human behavior, we have run a series of controlled laboratory experiments with human subjects [Pita *et al.*, 2009]. In these experiments, subjects play a "pirates and treasure" game designed to simulate an adversary planning an attack on an LAX

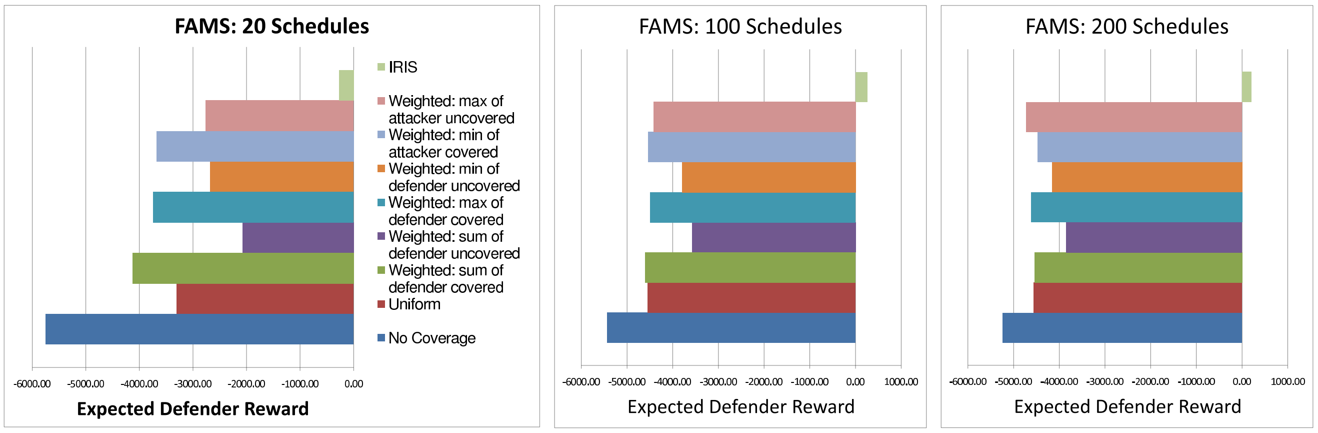
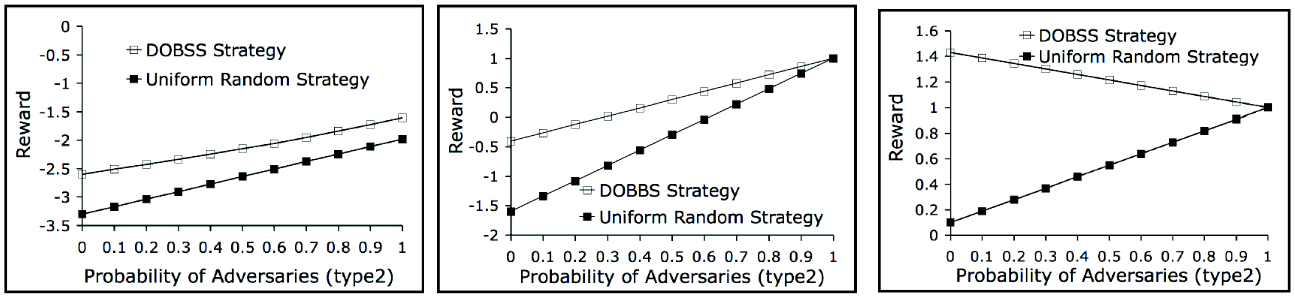


Figure 3: These graphs show the results of scheduling with the IRIS algorithm called ASPEN vs. other possible scheduling methods. In the case of (a) 20, (b) 100, and (c) 200 schedules, the IRIS scheduling method is superior. Results are averaged over 30 independent trials each.



(a) 1 checkpoint (b) 2 checkpoints (c) 3 checkpoints

Figure 4: This figure compares ARMOR's schedules (generated by the algorithm named DOBBS) with a uniform random baseline schedule. Figures a–c show the utility of schedules for 1–3 vehicle checkpoints varying the relative probability of two different attacker types. The x-axes show the probability of the two attacker types (where 0 corresponds to 0% attack type 2, and 100% attack type 1) and y-axes show the expected utility of ARMOR and a uniform random defense strategy.

terminal, shown in Figure 5. Subjects are given information about the payoffs for different actions and the pirates' strategy for defending their gold (analogous to the security policy for defending airport terminals). Subjects receive cash payments at the conclusion of the experiment, based on their performance in the game.

These experiments have provided additional support for quality of ARMOR's schedules against human opponents. First, they suggest that the assumptions imposed by the game-theoretic model are reasonable. Second, we have tested many conditions, varying both the payoff structure and the observation ability, ranging from no observation of the defense strategy to perfect observation. The results show that ARMOR's schedules achieve higher payoffs than the uniform random benchmark across all of the experimental conditions tested, often by a large margin.[2] These results demonstrate that ARMOR schedules outperform competing methods when playing against human adversaries.

[2]New defense strategies developed in this work show even better performance against some (suboptimal) human adversaries by explicitly exploiting the attacker's weaknesses.

4.4 Arrest Data

Ideally, we would be able to use data from the operation of a deployed system to provide further validation of the modeling assumptions. For example, in the case of ARMOR we might be interested in the number of attacks prevented by the system. Unfortunately, such data is very limited in the case of ARMOR; there have been 0 major attacks on the airport, but it is impossible to say how many attacks would have occurred *without* ARMOR.

We were able to obtain some data on the arrest record at ARMOR checkpoints. Though the use of such data has multiple caveats (see Section 5 for more discussion), it can still provide some useful information. We received summarized and actual reports from the LAX police regarding the number of violations that they detected at checkpoints in 2007, 2008 and January 2009. For example, we received the following report for January 2009:

1. January 3, 2009: Loaded 9mm pistol discovered

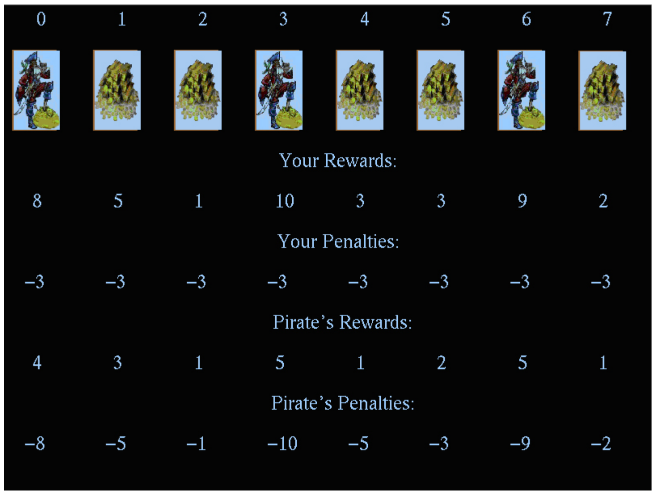


Figure 5: Screenshot of the "pirates and treasure" game

2. January 3, 2009: Loaded 9mm handgun discovered (no arrest)
3. January 9, 2009: 16 handguns, 4 rifles, 1 pistol, and 1 assault rifle discovered — some fully loaded
4. January 10, 2009: 2 unloaded shotguns discovered (no arrest)
5. January 12, 2009: Loaded 22cal rifle discovered
6. January 17, 2009: Loaded 9mm pistol discovered
7. January 22, 2009: Unloaded 9mm pistol discovered (no arrest)

Figure 6 tabulates the number of violations for the year prior to the deployment of ARMOR and during 2008 when ARMOR was in use. The x-axis breaks down the violations into different types and the y-axis represents the number of violations. The number of violations is substantially higher at LAX after ARMOR was deployed than in the preceding period. For example, only 4 drug related offenses were detected before the deployment of ARMOR while 30 such offenses were detected after the deployment. While we must be careful about drawing too many conclusion from this data due to the large number of uncontrolled variables (for example, the number of checkpoints was not consistent during this period), the ARMOR checkpoints do appear to be effective on this measure.

4.5 Qualitative Expert Evaluations

Given the sparseness and limitations of the available data from the field, evaluations of the system by security experts become a very important source of information. Though they are typically qualitative in nature, they are one of the few ways to gather evidence on the quality of the modeling assumptions and the effectiveness of the holistic system, as it is actually deployed and used on a day-to-day basis. Security procedures at LAX are subject to numerous internal and external security reviews (not all of which are public). The available qualitative reviews indicate ARMOR is both effective and highly visible.

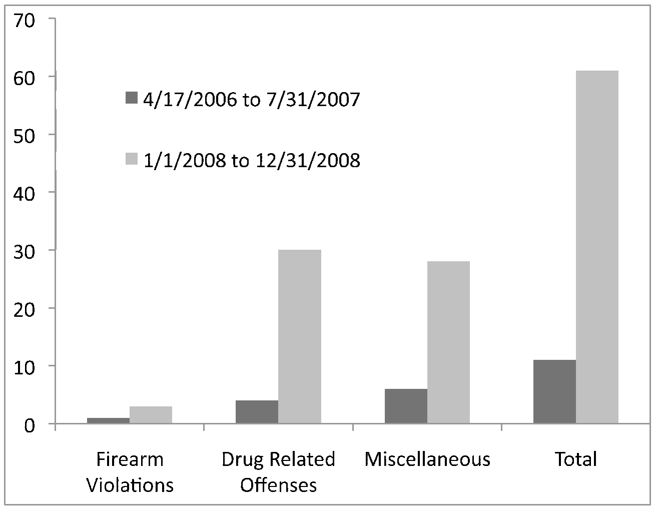


Figure 6: This figure shows how the number of arrests in three categories changed before and after the ARMOR deployment at LAX.

Director James Butts of the LAWA police reported that ARMOR "makes travelers safer" and even gives them "a greater feeling of police presence" [Murr, 2007]. Erroll Southers, Assistant Chief of LAWA police, told a Congressional hearing that "LAX is safer today than it was eighteen months ago," due in part to ARMOR. A recent external study by Israeli transportation security experts concluded that ARMOR was a key component of the LAX defensive setup. The ARMOR team has also been awarded Letters of Commendation from the city of Los Angeles in recognition of the efforts towards securing the Los Angeles International Airport. Thus, the domain experts have been highly supportive of ARMOR, and it would be very hard to deploy the system without their support. They are also likely to identify potential problems with the system quickly. While such studies are not very useful for quantifying ARMOR's benefit, they all suggest that the domain experts believe that ARMOR generates better schedules as compared to their previous approaches.

ARMOR was designed as a mixed initiative system that allows police to override the recommended policies. In practice, users have not chosen to modify the recommended schedules, suggesting that users are confident in the outputs. While such studies are not very useful for directly quantifying ARMOR's benefit, it would be very hard to deploy the system without the support of such experts. Furthermore, if there were an "obvious" problem with the system, such experts would likely identify it quickly.

Expert opinions have emphasized that an important benefit of the system is its transparency and visibility which contribute to deterrence. ARMOR assumes that adversaries are intelligent and have the ability to observe the security policy: knowing about the system does not reduce its effectiveness. The deployment of ARMOR has been quite visible: ARMOR has been covered on local TV stations (including FOX and NBC), in newspapers (including the LA Times and the International Herald Tribune), and in a national magazine

(Newsweek).

The IRIS system has been tested both qualitatively and quantitatively, but the results of these tests are restricted. However, James B. Curren, special assistant in the office of Flight Operations at the Federal Air Marshals Service, has affirmed the effectiveness of IRIS:

> We have tested IRIS and found it to be applicable to our needs in creating uncertainty as to FAM presence on selected city pairs of flights. After extensive testing we have implemented IRIS on a trial run of flight selections and have continued to expand the number of flights scheduled using IRIS. Our exact use of IRIS is sensitive information and we can only state that we are satisfied with IRIS and confident in using this scheduling approach.

Furthermore, internal governmental studies have both recommended that random scheduling be implemented, which is precisely what IRIS accomplishes [GAO, 2009][3]:

> Because the number of air marshals is less than the number of daily flights, FAMS's operational approach is to assign air marshals to selected flights it deems high risk–such as the nonstop, long-distance flights targeted on September 11, 2001. In assigning air marshals, FAMS seeks to maximize coverage of flights in 10 targeted high-risk categories, which are based on consideration of threats, vulnerabilities, and consequences. In July 2006, the Homeland Security Institute, a federally funded research and development center, independently assessed FAMS's operational approach and found it to be reasonable. However, the institute noted that certain types of flights were covered less often than others. The institute recommended that FAMS increase randomness or unpredictability in selecting flights and otherwise diversify the coverage of flights within the various risk categories. As of October 2008, FAMS had taken actions (or had ongoing efforts) to implement the Homeland Security Institute's recommendations. GAO found the institute's evaluation methodology to be reasonable.

5 Goals for Security Decision Support Systems

Any security system can be evaluated using a wide variety of metrics, including both costs and benefits. For the purposes of analysis, we divide benefits into two categories: *direct* and *indirect*. Direct benefits speak to benefits which may be measured, such as

- reduced security costs,
- attacks prevented or mitigated during execution,
- increased numbers of attackers caught, or
- reduced damage from successful attacks.

In contrast, indirect benefits include

- attacks prevented through deterrence,
- increased attacker planning time,
- increased requirements for a successful attack,
- improved public perceptions of security, or
- improved (qualitative) assessments of security by experts.

Regardless of how such benefits are partitioned, some are easier to directly measure than others. However, all of them speak to the idea of defender utility: given a finite amount of resources, how can the defender maximize security per dollar spent? As discussed previously, no security system can be 100% effective — there is always a chance that given enough resources and planning, and adversary may compromise a system. A system may, however, require higher amounts of equipment, manpower, and/or planning in order to compromise it, relative to having the system disabled. Thus the question becomes not "is the system effective," but "is system A more effective than system B," or "how much does the system improve defensive capabilities per dollar spent?" We next elaborate upon two of these themes: utility and deterrence.

5.1 Security per Dollar

Evaluating security systems requires a full cost-benefit analysis that takes a comprehensive view of both the benefits and the costs of the system. There is always additional defensive capability that could be purchased — another guard could be hired, one more piece of technology could be installed or upgraded, etc. Of course, additional security typically comes with diminishing returns, and there is a policy decision to be made about how many resources to devote to security. Improving the efficiency of security can be used to either 1) increase defensive abilities on a fixed budget, and/or 2) decrease expenditure for a fixed defensive capability. Such increases in efficiency are the primary goal of decision support systems like ARMOR and IRIS.

Recall that Figure 6 showed how the defender's utility changed with different numbers of K9 units. Such a graph of the trade-off between defensive capability and cost can allow policy makers to more easily see the trade-off between money and security when using the ARMOR system. More important, one can easily see how much ARMOR helps increase security relative to uniform random (or other randomization strategies), which is a critical factor when deciding whether or not to implement a given security technique. It directly follows that countermeasure investments with the highest cost-effectiveness ratio should be implemented first, assuming that there is no inter-dependence between proposed measures. [Edmunds and Wheeler, 2009] Determining exactly how much safer ARMOR makes the airport is a very difficult question, but as we will discuss in the following sections, arguing that ARMOR improves safety on a per-office basis (and thus "safety per dollar" spent) is not difficult.

5.2 Threat Deterrence

A key goal of many security systems is *deterrence*: an effective system will not only identify and prevent successful

[3]This report is a public version of the restricted report GAO-09-53SU.

attacks, but will also dissuade potential attackers. Unfortunately, it is typically impossible to directly measure the deterrence effect. To measure deterrence directly, one needs to know how many attacks *did not occur* due to security, a generally unmeasurable counterfactual.

In all cases relevant to this chapter, we assume that the attacker is adaptive, and does not act blindly, but takes the defender's actions into account. It is only through this adaptability that defensive measures could deter an attacker. For instance, one could model the uncertainty due to chance or attacker choices via an *event tree model* or a *decision tree model* [Edmunds and Wheeler, 2009]. We choose to focus on game theoretic methods as they have strong mathematical underpinnings and are ideally suited to reasoning about adaptable, rational, actors. Game models naturally factor in deterrence, in that an intelligent attacker is assumed to switch to a different strategy if the defender uses security resources to make the original strategy undesirable.

How to best understand and accurately model deterrence is a topic of current research in the threat assessment community [Taquechel, 2010]. For instance, for a threat (i.e., a defensive measure) to be an effective deterrent, it must be *credible* in the eyes of the attacker [Berejikian, 2002], which may involve complex *signalling* effects between the parties [Gerson and Boyars, 2007]. As mentioned above, ARMOR has been well-publicized in the popular press. While there are many other less public security measures employed at LAX, ARMOR may lead a potential attacker to decide not to attack LAX and instead stay at home and not attack. Another possible outcome of ARMOR's publicity is that an attacker select a more vulnerable target. While this is a win from the standpoint of LAX, it may or may not be an improvement for Los Angeles, the state of California, or the United States, depending on the target selected. However, if LAX is not targeted due to deterrence measures, the eventual target is presumably less attractive to the attackers, resulting in a net increase in the defender's utility.

6 Types of Evaluation

This section of the chapter ties together the discussion of the previous sections, showing how the different evaluations performed on ARMOR and IRIS help to verify that the goals of the security system are achieved.

6.1 Model-Based/Algorithmic

The first type of evaluations (and the most natural for computer scientists) are based on analysis of the model and the underlying algorithms. Given assumptions about the attacker (e.g., the payoff matrix is known), game theoretic tools can be used to determine the attacker's expected payoff. Additionally, deterrence can be measured by including a "stay home" action, returning neutral reward. These types of analyses speak to such concerns.

First, the sensitivity of the solution method can be examined. No model will be perfect, but a solution to that model would ideally be robust to small imperfections in the model. This type of analysis is able to speak to the potential impact of different kinds of abstraction and modeling error on the final solution.

Second, the utility of the solution method can be directly estimated from algorithmic evaluations. For instance, the attacker resources vs. expected attacker utility and the defender resources vs. expected defender utility can both be estimated. To the extend that the model is correct, these utilities can be determined exactly, providing an important tool to both policy makers and security professionals when deciding what defensive measures to implement.

Finally, the underlying algorithms can be evaluated, both to prove correctness (or a close approximation), and to evaluated the computational effort required to compute a solution for problems with different sizes and properties.

6.2 Cost/Benefit Analysis

Another type of analysis, which can combine both algorithmic results and those from the implementation, is a full cost/benefit analysis of the deployed system. For instance, the cost of implementation and maintenance on the system can be directly measured. The benefit of the system can be measured both by changes in the defenders' utility (e.g., Figure 2) and in terms of less tangible factors (such as such as quantifying increases in travel time or a decrease in civil liberties).

A system-wide cost/benefit analysis can help security professionals decide where to allocate finite resources in order to best protect the entire area by changing staffing levels and/or implementing new security measures. Additionally, such studies can provide important information to other sites that may consider deploying a new security measure, such as ARMOR. The ultimate goal of this analysis is to answer the questions "how useful is this defensive measure?" and "how does utility change for different levels of defender / attacker resources?"

6.3 Relative Benefit

Measuring the *relative* benefit of defensive measures is similar to measuring the absolute utility, as described in the previous section. However, an important difference is that such measurements may be less brittle to model inaccuracies. For instance, when deciding whether or not to implement ARMOR, a site may consider the absolute utility of ARMOR using a cost/benefit analysis, which depends on the assumptions about the attacker. A relatively simple analysis could instead look at how utility will be changed by a proposed security method.

For instance, Figure 1 shows how the utility of ARMOR compares to different scheduling strategies. A relative benefit analysis makes it easier to answer the question "should this measure be implemented?" and may be particularly useful when discussing security with professionals who are leery of implementing decision-theoretic methods.

6.4 Human Behavioral Experiments

Human psychological studies can help to better simulate attackers in the real world (e.g., Figure 5). Evaluations on an abstract version of the game may test base assumptions, or a detailed rendition of the target in a virtual reality setting with physiological stress factors could test situated behavior. Human subjects may allow researchers to better simulate the

Summary of Evaluation Types

Evaluation	Problem Formulation			Goal Accomplishments	
	Abstraction	Solution	Implementation	Direct Benefit	Indirect Benefit
Algorithmic	✓	✓		✓	✓
Cost/Benefit			✓	✓	
Relative Benefit			✓	✓	
Human Experiments	✓		✓	✓	
Operational Record			✓	✓	
High Level Evaluations	✓		✓	✓	✓

Table 1: This table summarizes the types of experiments we have conducted on our security systems and show how they help verify the problem formulation and/or that the system is accomplishing its goals.

actions of attackers, who may not be fully rational. Human tests suffer from the fact that participants are not drawn from the same population as the actual attackers (i.e., undergraduate college students).

In some situations, it may be possible to conduct realistic human studies in the true setting, i.e., employ a "red team." Such tests can use qualified security personnel attempt to probe security defenses provides realistic information in life-like situations using the true defenses (including those that are not visible). However, such a test is very difficult to conduct as some security must be alerted (so that the team is not endangered) while remaining realistic, the tests are often not repeatable, and a single test is likely unrepresentative. To the extent that such tests are feasible, they would speak to both the implementation of the system and how well it accomplishes its direct goals.

6.5 Operational Record

Analysis of the operational record of a system provides confirmation that the system works, but is not as useful as one may initially think. Ideally, the system would be enabled and disabled for randomized periods of time, allowing for a careful study of the system. Unfortunately, this is typically infeasible.

Consider the ARMOR system: that there are arrests resulting from the system shows that the system does allow rule breakers to be caught. However, comparing the number of arrests before and after ARMOR may or may not be useful. For instance, if the number of arrests increases, this could be an indication that ARMOR is more effective than the previous system. However, it could also mean that more people are violating the rules during this time period, and in fact a lower number of criminals are caught. In the opposite case, a lower number of arrests could mean that deterrence from ARMOR is convincing more criminals to stay away. Or, a lower number of arrests could mean that more people are circumventing the system. Without knowing the number of criminals that are not caught, it is impossible to tell how well a particular piece of the security is performing.

An additional complication is that, thankfully, most security threats are very low-frequency events. This means that data is collected relatively infrequently, making analysis of the operational record more sparse than in other (non-security) deployed applications.

6.6 High-level Evaluations

While computer scientists traditionally prioritize precise, repeatable studies, this is not always possible in the security community; computer scientists are used to quantitative evaluations in controlled studies, whereas security specialists are more accepting of qualitative metrics on deployed systems. For instance, Lazaric [1999] summarized a multi-year airport security initiative by the FAA where the highest ranked evaluation methodology (of seven) relied on averaging *qualitative* expert evaluations.

Assuming the high-level evaluation is appropriately done, it may address the abstraction and implementation questions arising from problem formulation, as well as address both direct and indirect goals. While these studies generally do not produce quantitative numbers that can be used to determine the utility of a security measure, they may uncover flaws in that measure, or in the security of the entire system as a whole. If no such flaws are found, the study may support the hypothesis that the security measure in question does improve overall security. As with the other types of evaluation, the goal is not to prove that the system works, but to provide evidence towards this effect.

7 Related Work

Security is a complex research area, spanning many disciplines, and policy evaluation is a persistent challenge. Many security applications are evaluated primarily on the basis of theoretical results; situated evaluations and even laboratory experiments with human subjects are relatively rare. In addition, existing general methodologies for risk and security evaluation often rely heavily on expert opinions and qualitative evaluations.

Lazarick [1999] is a representative example which relies heavily on expert opinions. In the study, seven tools/approaches used to evaluate airport security were compared as part of a competitive bidding process. At the end of the multi-year security initiative, the highest ranked evaluation methodology relied on averaging qualitative expert evaluations.

A second example of a high-level methodology for per-facility and regional risk assessment, such as described by Baker [2005]. The methodology relies heavily on expert opinions and evaluations from local technical staff/experts, similar to Lazarick [1999]. The three key questions in the methodology are: (1) Based on the vulnerabilities identified, what is the likelihood that the system will fail? (2) What are the consequences of such failure (e.g., cost or lives)? (3) Are these consequences acceptable? Such an approach enumerates all vulnerabilities and threats in an attempt to determine what should (or must) be improved. There is no quantitative framework for evaluating risk.

Many in the risk analysis community have recently argued for game theory as a paradigm for security evaluation, with the major advantage that it explicitly models the adaptive behavior of an intelligent adversary. Cox [2008] provides a detailed discussion of the common "Risk = Threat $\times$ Vulnerability $\times$ Consequence" model, including analysis of an example use of the model. There are several arguments raised as weaknesses of the approach, including (1) the values are fundamentally subjective (2) rankings of risk are often used, but are insufficient (3) there are mathematical difficulties with the equation, including dependencies between the multiplied terms, and (4) the model does not account for adaptive, intelligent attackers. One of the main recommendations of the paper is to adopt more intelligent models of attacker behavior, instead of more simple, static, risk estimates.

Bier et al. [2009] provide a high-level discussion of game-theoretic analysis in security applications and their limitations. The main argument is that the *adaptive* nature of the terrorist threat leads to many problems with static models — such models may overstate the protective value of a policy by not anticipating an attacker's options to circumvent the policy. They explicitly propose using quantitative risk analysis to provide probability/consequence numbers for game-theoretic analysis.

Beir [2007] performs a theoretical analysis of the implications of a Bayesian Stackelberg security game very similar to the one solved by ARMOR, although most of the analysis assumes that the defender does *not* know the attacker's payoffs. The primary goal is to examine intuitive implications of the model, such as the need to leave targets uncovered in some cases so as not to drive attackers towards more valuable targets. There are no "real world" evaluation of the model. Other work [Bier *et al.*, 2008] considers high-level budget allocation (e.g., to large metropolitan areas). While the study uses real data, its focus is not model evaluation but the implications resulting from the model.

Game theory does have much to offer in our view, but should not be considered a panacea for security evaluation. One difficulty is that human behavior often does not correspond exactly to game-theoretic predictions in controlled studies. Weibull [2004] describes many of the complex issues associated with testing game-theoretic predictions in a laboratory setting, including a discussion of the ongoing argument regarding whether people typically play the Nash equilibrium or not (a point discussed at length in the literature, such as in Erev et al. [2002]). This is one reason we believe behavioral studies with humans are an important element for security system evaluation.

Many of the issues we describe in acquiring useful real-world data for evaluation purposes are mirrored in other types of domains. Blundell and Costa-Dias [2009] describe approaches for experimental design and analysis of policy proposals in microeconomics, where data is limited in many of the same ways: it is often not possible to run controlled experiments and many desired data cannot be observed. They describe several classes of statistical methods for these cases, some of which may be valuable in the security setting (though data sensitivity and sparse observations pose significant additional challenges). In truth, it is often hard to evaluate complex deployed systems in general — in our field a test of the prototype often suffices (c.f., Scerri et al. [2008]).

Jacobson et al. [2005] describe a deployed model for screening airline passenger baggage. The model includes detailed information regarding estimated costs of many aspects of the screening process, including variables for probability of attack and cost of a failed detection, but these are noted to be difficult to estimate and left to other security experts to determine. One particularly interesting aspect of the approach is that they perform sensitivity analysis on the model in order to assess the effect of different values on the overall decisions. Unfortunately, the authors have little to say about actually setting the input values to their model; in fact, there is no empirical data validating their screening approach.

Kearns and Ortiz [2003] introduce algorithms for a class of "interdependent" security games, where the security investment of one player has a positive externality and increases the security of other players. They run the algorithms on data from the airline domain but do not directly evaluate their approach, instead looking at properties of the equilibrium solution and considering the broad insight that this solution yields regarding the benefits of subsidizing security in such games.

Lastly, the field of *fraud detection* [Kou *et al.*, 2004], encompassing credit card fraud, computer intrusion, and telecommunications fraud, is also related. Similar to the physical security problem, data is difficult to access, researchers often do not share techniques, and deterrence is difficult (or impossible) to measure. Significant differences include:

1. Humans can often classify (in retrospect) false positives and false negatives, allowing researchers to accurately evaluate strategies.
2. Companies have significant amounts of data regarding known attacks, even if they do not typically share the data outside the company. Some datasets do exist for common comparisons (c.f., the 1998 DARPA Intrusion Detection Evaluation data[4]).
3. The frequency of such attacks is much higher than physical terrorist attacks, providing significant training/evaluation data.
4. Defenders can evaluate multiple strategies (e.g., classifiers) on real-time data, whereas physical security may employ only, and evaluate, one strategy at a time.

8 Conclusions

This chapter has discussed existing evaluations of the deployed ARMOR and IRIS systems. These results show how such systems can be reasonably evaluated, and in particular

[4]For data and program details, see `http://www.ll.mit.edu/mission/communications/ist/index.html`.

show that ARMOR works well in theory, and that security experts agree it is beneficial. In many ways, this level of evaluation goes beyond what is typical of applications, even those deployed in real-world settings. Overall, we find strong evidence to support the use of ARMOR over previous methods.

Another point worth stressing is that ARMOR and IRIS are relatively easy to use. ARMOR in particular has been instrumental in aiding the police forces to efficiently and more conveniently generate schedules to deploy more and more units. For example, consider a situation when only 2 canines need to be scheduled for 2 hours each over any of the 7 terminals. Each canine could be assigned to any of the 7 terminals each hour, making the search space as large as $7^4 (= 2401)$ combinations. This search space grows exponentially with the number of canines and the number of hours for which the schedule needs to be generated, making it impractical for human schedulers. Thus, ARMOR has played a significant role in reducing, if not completely eliminating, the work of officers who manually constructed patrolling schedules. Additionally, the use of ARMOR has also made it possible for the security officers to update the generated schedules, in case more resources become available or new constraints need to be incorporated. Furthermore, even though ARMOR was designed as a mixed initiative system, users have chosen not to modify ARMOR schedules in practice, which suggests that the output schedules are indeed high-quality, and that domain experts have not chosen to 'tweak' the system's decisions. These added benefits have themselves been a contributing factor towards the continued use of schedules generated by ARMOR. Most importantly, when considering the cost of implementing a decision support system, it is important to consider ways in which the system may actually reduce security costs.

While none of the evaluation tests presented can calculate a measure's utility with absolute accuracy, understanding what each test *can* provide will help evaluators better understand what tests *should* be run on deployed systems. The goal of such tests will always be to provide better understanding to the "customer," be it researchers, users, or policy makers. By running multiple types of tests, utility (the primary quantity) can be approximated with increasing reliability.

At a higher level, thorough cost-benefit analyses can provide information to policy makers at the inter-domain level. For instance, consider the following example from Tengs and Graham [1996]:

> To regulate the flammability of children's clothing we spend \$1.5 million per year of life saved, while some 30% of those children live in homes without smoke alarms, an investment that costs about \$200,000 per year of life saved.

While such a comparative cost-benefit analysis is beyond the scope of the current study, these statistics show how such an analysis can be used to compare how effective measures are across very different domains, and could be used to compare different proposed security measures.

In the future we plan to use this framework to help decide which evaluation tests are most important to determining the utility of a deployed, security-focused decision support system. Additionally, we intend to continue collaborating with security experts to determine if our evaluations are sufficiently general to cover all existing types of security tests.

Currently, the majority of our evaluations have been conducted on the ARMOR system. However, we intend to continue testing IRIS and other newly-developed domains, to both better evaluate the domains, and to attempt to discover or improve evaluation techniques in the context of deployed systems.

Finally, a new type of decision support system is currently being developed at the TEAMCORE lab, which will focus on scheduling patrols on the LA subway system to look for ticketless travelers. While this application can still be framed as a security problem, we expect that there will be much more data available. In particular, we will both have more events when "attackers" are caught using the subway system without a ticket and we may be able to enable and disable the system on different days. An additional attractive option is that we may be able to occasionally have 100% coverage, allowing us to measure the ground truth, and how the deterrence effect impacts the number of ticketless travelers over time.

Acknowledgements

The authors would like to thank Manish Jain, James Pita, and Vicki Bier for helpful comments and suggestions, Sergeant Jim Holcomb for supplying ARMOR-related arrest data, and First Sargent Cruz for discussing pre-ARMOR scheduling procedures. This research was supported in part by the United States Department of Homeland Security through the National Center for Risk and Economic Analysis of Terrorism Events (CREATE) under grant number 2007-ST-061-000001. However, any opinions, findings, and conclusions or recommendations in this document are those of the authors and do not necessarily reflect views of the United States Department of Homeland Security.

References

[Baker, 2005] G. H. Baker. A vulnerability assessment methodology for critical infrastructure sites. In *DHS Symposium: R and D Partnerships in Homeland Security*, 2005.

[Berejikian, 2002] J. Berejikian. A cognitive theory of deterrence. *Journal of Peace Research*, (39):165–183, 2002.

[Bier *et al.*, 2008] V. M. Bier, N. Haphuriwat, J. Menoyo, R. Zimmerman, and A. M. Culpen. Optimal resource allocation for defense of targets based on differing measures of attractiveness. *Risk Analysis*, 28(3):763–770, 2008.

[Bier *et al.*, 2009] V. M. Bier, Jr. L. A. Cox, and M. N. Azaiez. Why both game theory and reliability theory are important in defending infrastructure against intelligent attacks. In *Game Theoretic Risk Analysis and Security Theats*, volume 128. Springer US, 2009.

[Bier, 2007] V. M. Bier. Choosing what to protect. *Risk Analysis*, 27(3):607–620, 2007.

[Blundell and Costa-Dias, 2009] R. Blundell and M. Costa-Dias. Alternative approaches to evaluation in empirical microeconomics. *Journal of Human Resources*, 2009.

[Conitzer and Sandholm, 2006] V. Conitzer and T. Sandholm. Computing the optimal strategy to commit to. In *Proc. of EC*, 2006.

[Cruz, 2009] First Sargent Cruz. Personal communication, August 20 2009.

[Edmunds and Wheeler, 2009] Thomas Edmunds and Richard Wheeler. Setting priorities. In Stephen M. Maurer, editor, *WMD Terrorism*, chapter 7, pages 191–209. MIT Press, Cambridge, MA, 2009.

[Erev *et al.*, 2002] I. Erev, A. E. Roth, R. L. Slonim, and G. Barron. Predictive value and usefulness of game theoretic models. *International Journal of Forecasting*, 18(3):359–368, 2002.

[GAO, 2009] The United States Government Accountability Office: GAO. Aviation security: Federal air marshal service has taken actions to fulfill its core mission and address workforce issues, but additional actions are needed to improve workforce survey, January 2009. GAO-09-273.

[Gerson and Boyars, 2007] M. Gerson and J. Boyars. The future of U.S. deterrence; constructing effective strategies to deter states and non-state actors, 2007. `www.cna.org/documents/D0017171.A2.pdf`.

[Jacobson *et al.*, 2005] S. H. Jacobson, T. Karnai, and J. E. Kobza. Assessing the impact of deterrence on aviation checked baggage screening strategies. *International J. of Risk Assessment and Management*, 5(1):1–15, 2005.

[Kearns and Ortiz, 2003] M. Kearns and L. E. Ortiz. Algorithms for interdependent security games. In *Neural Information Processing Systems (NIPS)*, 2003.

[Kiekintveld and Wellman, 2008] Christopher Kiekintveld and Michael Wellman. Selecting strategies using empirical game models: An experimental analysis of meta-strategies. In *AAMAS-08*, 2008.

[Kiekintveld *et al.*, 2009] Christopher Kiekintveld, Manish Jain, Jason Tsai, James Pita, Fernando Ordónez, and Milind Tambe. Computing optimal randomized resource allocations for massive security games. In *AAMAS*, 2009.

[Kiekintveld *et al.*, 2011] Christopher Kiekintveld, Janusz Marecki, and Milind Tambe. Approximation methods for infinite Bayesian Stackelberg games: Modeling distributional payoff uncertainty. In *AAMAS-11*, 2011.

[Kou *et al.*, 2004] Y. Kou, C. Lu, S. Sinvongwattana, and Y.P. Huang. Survey of fraud detection techniques. In *Proc. of IEEE Networking*, 2004.

[L. A. Cox, 2008] Jr. L. A. Cox. Some limitations of "risk = threat x vulnerability x consequence" for risk analysis of terrorist attacks. *Risk Analysis*, 28(6):1749–1761, 2008.

[Lazarick, 1999] R. Lazarick. Airport vulnerability assessment – a methodology evaluation. In *Proc. of 33rd IEEE International Carnahan Conference on Security Technology*, 1999.

[Murr, 2007] A. Murr. Random checks. In *Newsweek National News*. September 2007. http://www.newsweek.com/id/43401.

[Paruchuri *et al.*, 2008] Praveen Paruchuri, Jonathan P. Pearce, Janusz Marecki, Milind Tambe, Fernando Ordonez, and Sarit Kraus. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *AAMAS-08*, 2008.

[Pita *et al.*, 2008] J. Pita, M. Jain, C. Western, C. Portway, M. Tambe, F. Ordonez, S. Kraus, and P. Paruchuri. Deployed ARMOR protection: The application of a game theoretic model for security at the Los Angeles International Airport. In *Proc. of AAMAS*, 2008.

[Pita *et al.*, 2009] J. Pita, M. Jain, M. Tambe, F. Ordonez, S. Kraus, and R. Magori-Cohen. Effective solutions for real-world stackelberg games: When agents must deal with human uncertainties. In *Proc. of AAMAS*, 2009.

[Scerri *et al.*, 2008] P. Scerri, T. Von Goten, J. Fudge, S. Owens, and K. Sycara. Transitioning multiagent technology to UAV applications. In *Proc. of AAMAS Industry Track*, 2008.

[Stevens *et al.*, 2009] D. Stevens, T. Hamilton, M. Schaffer, D. Dunham-Scott, J. J. Medby, E. W. Chan, J. Gibson, M. Eisman, R. Mesic, C. T. Kelly, J. Kim, T. LaTourrette, and K. J. Riley. Implementing security improvement options at Los Angeles international airport, 2009. `www.rand.org/pubs/documented_briefings/2006/RAND_DB499-1.pdf`.

[Taquechel, 2010] Eric F. Taquechel. Validation of rational deterrence theory: Analysis of U.S. government and adversary risk propensity and relative emphasis on gain or loss. Master's thesis, Naval Postgraduate School, March 2010.

[Tengs and Graham, 1996] T. O. Tengs and J. D. Graham. Risks, costs, and lives saved: Getting better results from regulation. In *The opportunity costs of haphazard social investments in lifesaving*. American Enterprise Institute, Washington, 1996.

[Tsai *et al.*, 2009] Jason Tsai, Shyamsunder Rathi, Christopher Kiekintveld, Fernando Ordóñez, and Milind Tambe. IRIS - A tools for strategic security allocation in transportation networks. In *AAMAS (Industry Track)*, 2009.

[Weibull, 2004] J. Weibull. Testing game theory. In Steffen Huck, editor, *Advances in Understanding Strategic Behavior: Game Theory, Experiments and Bounded Rationality.*, pages 85–104. Palgrave MacMillan, 2004.