Solving Non-Zero Sum Multiagent Network Flow Security Games with Attack Costs

Steven Okamoto
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
sokamoto@cs.cmu.edu

Noam Hazon
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
noamh@cs.cmu.edu

Katia Sycara
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA 15213
katia@cs.cmu.edu

ABSTRACT

Moving assets through a transportation network is a crucial challenge in hostile environments such as future battlefields where malicious adversaries have strong incentives to attack vulnerable patrols and supply convoys. Intelligent agents must balance network costs with the harm that can be inflicted by adversaries who are in turn acting rationally to maximize harm while trading off against their own costs to attack. Furthermore, agents must choose their strategies even without full knowledge of their adversaries' capabilities, costs, or incentives.

In this paper we model this problem as a non-zero sum game between two players, a sender who chooses flows through the network and an adversary who chooses attacks on the network. We advance the state of the art by: (1) moving beyond the zero-sum games previously considered to non-zero sum games where the adversary incurs attack costs that are not incorporated into the payoff of the sender; (2) introducing a refinement of the Stackelberg equilibrium that is more appropriate to network security games than previous solution concepts; and (3) using Bayesian games where the sender is uncertain of the capabilities, payoffs, and costs of the adversary. We provide polynomial time algorithms for finding equilibria in each of these cases. We also show how our approach can be used for games where there are multiple adversaries.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Intelligent agents, Multiagent systems*; J.4 [**Social and Behaviorial Sciences**]: [Economics]

General Terms

Theory, Security, Economics

Keywords

network security game, communication security, multiagent communication

1. INTRODUCTION

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, Conitzer, Winikoff, Padgham, and van der Hoek (eds.), June, 4–8, 2012, Valencia, Spain.

Many multiagent applications must utilize networks in inherently hostile environments in which adversaries have strong incentives to disrupt operations, as when enemies attack vulnerable patrols and supply convoys in transportation networks using asymmetric warfare techniques. Any multiagent deployment in these environments must address the crucial issue of strategically moving assets in a secure and effective manner in the presence of such malicious adversaries. Game theory offers a natural and rigorous framework for reasoning strategically in these kinds of adversarial domains.

Such hostile network environments share six key characteristics: **(1)** The topology of the network creates exponential sized strategy spaces that cannot be solved efficiently using standard normal form techniques. **(2)** Security is not the sole criterion but must be balanced with competing performance objectives. For example, shorter paths are preferred by supply convoys to minimize fuel costs and by sensor networks to conserve battery power and reduce latency. **(3)** Patterns of behavior may be learned by the adversary. For example, the adversary may observe supply convoys in secret before planning and executing his attack. **(4)** Adversaries are rational agents who balance the harm that they can inflict with costs of attacking. **(5)** Information on the adversary's capabilities, payoffs, and costs is rarely available, and estimates must be used instead. **(6)** Multiple adversaries with differing abilities may be present.

Work in network security games have addressed the first [13, 10, 7], second [10], and third [13, 7] of these, but have left the others largely untouched. Attack costs have been ignored by assuming a zero-sum payoff structure, so that the the incentives of the sender and adversary are exactly and oppositely aligned. In general this leads to a computationally simpler problem, but does not reflect the fact that the attack costs do not factor in to the payoff of the sender. Abandoning the zero-sum assumption is also essential to addressing the fourth point because it is known that in zero-sum games it does not matter if the adversary can observe the behavior patterns before choosing a strategy [15]. In this paper we address each of these characteristics, starting most importantly by allowing non-zero sum payoffs based on attack costs.

We model the problem as a game between a sender and an adversary. The sender chooses a flow through the network from the source nodes to the sink. The adversary chooses one or more attacks from a set of possible attacks, where each attack adds penalties to one or more link in the network. For example, one attack may be to jam a node in a communication network, thereby interfering with the communication with that targeted node and also (although to a lesser extent) to all communication with neighboring nodes as well. When the sender utilizes a link that has been attacked, he suffers harm proportional to the total penalty on the link and the amount of flow being sent on the link. The adversary incurs a cost for each attack, and different attacks may have different costs reflecting the differing degrees of difficulty or ease of attack. The sender seeks to minimize harm while the attacker seeks to maximize the harm minus the attack costs.

In this paper we advance the state of the art by combining the existing approaches with three major new contributions. First, we assume non-zero sum payoffs due to attack costs that adversary incurs in attacking the network. These costs factor into his payoff but not the payoff of the sender. We provide polynomial time algorithms based on linear programs (LPs) for finding Nash equilibria in these games. Second, the non-zero sum payoffs allow the possibility of the sender improving his payoff by committing to strategies in a Stackelberg game. We show that the existing solution concepts are inappropriate for network security games and introduce a refinement of the Stackelberg equilibrium based on the sender's ability to affect the adversary's strategy by deviating from equilibrium behavior. Finally, we consider games of incomplete information where the sender knows only probability distributions over the maximum number of nodes that the adversary can attack, and the adversary's payoffs and costs. We formulate these as Bayesian games and provide polynomial time algorithms for finding equilibria. We also show how this approach can be generalized to model games with multiple adversaries.

2. SIMULTANEOUS GAME WITH ATTACK COSTS

2.1 Model

We start with the network flow security game with attack costs but no uncertainty. This game is played between a sender and an adversary taking actions on a network represented by a directed graph $G = (V, E)$ with $n = |V|$ nodes and $m = |E|$ edges. The sender chooses how to send flow from a set $S \subset V$ of *source nodes* to the sink node $t \in V$. The amount of flow originating at a node $v \in V$ that must be sent to t is denoted by b_v, with $b_v > 0$ for all $v \in S$ and $b_v = 0$ for all $v \notin S$. The sender's strategy space $\mathcal{F}$ is the set of all feasible flows from the source nodes to the sink. Flows may be divided on alternate paths from the source nodes to the sink[1], leading to a continuous strategy space for the sender if there are at least two paths from any source node to the sink. A sender strategy f is represented as a $m \times 1$ vector where f_e is the amount of flow sent on edge $e \in E$. For convenience, for an edge $(u, v) \in E$, $f_{(u,v)}$ is denoted simply as f_{uv}.

The adversary chooses attacks from a set A. The adversary has a cost for each attack, represented by the $|A| \times 1$ attack cost vector c, where $c_a \geq 0$ is the the cost suffered by the adversary for attack $a \in A$. The adversary can execute up to k attacks simultaneously. Thus the adversary's set of pure strategies $\mathcal{A}$ is the set of all subsets of A of size at most k, which has size $\Theta(|A|^k)$, and his set of mixed strategies is the set of all probability distributions over $\mathcal{A}$. Instead of representing mixed strategies explicitly, we use the marginal probability distribution represented by the $1 \times |A|$ vector p, where p_a is the marginal probability of the adversary executing $a \in A$. This is sufficient for computing payoffs (and hence equilibrium behavior) [10], and so we sometimes refer to p as the adversary's mixed strategy. Because of the size of $\mathcal{A}$, even describing a mixed strategy explicitly requires exponential time in general, but it is possible to efficiently sample a pure strategy in conformance with p using algorithms such as comb sampling [13] or weighted random sampling [4].

The payoff for the sender in the game is quantified by the *harm* suffered as a result of the adversary's attacks. The harm is represented by a *harm matrix* M with $|A|$ rows and m columns, where each row specifies the penalties on edges caused by an attack so that entry M_{ij} is the per-unit-flow harm suffered when the adversary executes attack a_i and and the sender transmits flow on edge e_j. Harm for multiple attacks is summed, as occurs when a convoy must endure multiple attacks on its route, or when multiple jamming attacks in different parts of an ad hoc network additively increase the latency of messages. This representation models a broad range of harm functions that cannot be represented in other network security games [10]. When the sender plays f and the adversary plays p, the total expected harm is pMf and so the sender's payoff is $-pMf$. The payoff for the adversary depends on the harm that the sender suffers and the cost of the attacks, computed as $pMf - pc$. We refer to the adversary's payoff as his *reward*.

When the players choose their actions without any observations of the other player the game is played as a simultaneous move game and we use the familiar Nash equilibrium solution concept. A strategy profile (f^*, p^*) is a Nash equilibrium if f^* is a best response to p^* (for the sender) and p^* is a best response to f^* (for the adversary). In equilibrium, neither player has incentive to deviate and hence both are indifferent between their possible strategies.

To illustrate the importance of the attack costs on the Nash equilibrium, consider the network in Figure 1. Assume that $b_s = 1$ and $k = 1$ and that the adversary can choose to attack the top path or the bottom path with harm matrix

$$M = \begin{bmatrix} 102 & 0 & 0 & 0 \\ 0 & 3 & 0 & 0 \end{bmatrix}$$

Let f_i denote the amount of flow on edge e_i and note that the sender's strategy is fully specified by f_1 as $f_2 = 1 - f_1$. Let p_1 and p_2 denote the probability of attacking the top and bottom paths respectively.

When attack costs are zero, the adversary never has incentive not to attack, so $p_1 + p_2 = 1$. In equilibrium the adversary is indifferent between the two attacks so $102 f_1 = 3(1 - f_1)$ and so $f_1 = 3/105$. Similarly, the sender is indifferent between the two paths so $102 p_1 = 3(1 - p_1)$ and so $p_1 = 3/105$. An intuitive interpretation is that the sender sends most of his flow on the bottom path because of the lower potential for harm, while the adversary, being able to deduce this, attacks the bottom path with high probability because that's where most of the flow is.

[1]This approach can be used even when the asset moving through the network cannot be split, as with a convoy that must travel intact. The flow is then an efficient polynomial-sized representation for a mixed strategy over the exponential number of paths from the source nodes to the sink. Splits in the flow correspond to randomization over possible paths.

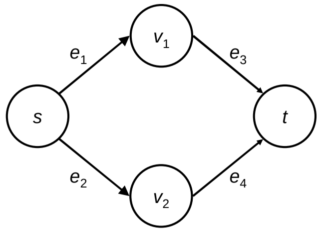

Figure 1: An example of a network with two possible paths.

Now suppose that attacking the top path has a cost $c_1 = 100$. The adversary's equilibrium strategy remains the same because the sender's payoff hasn't changed, but now for the adversary to be indifferent it must be that $102f_1 - 100 = 3f_2$ so that $f_1 = 103/105$. An intuitive explanation is that the adversary attacks the top path with low probability because of the high attack cost, and the sender, deducing this, sends most of the flow on the top path despite the high potential for harm because the adversary is unlikely to attack there.

The attack costs can also cause the adversary to not execute his maximum number of attacks because the cost outweighs the harm. For example, if $c_1 > 102$ then the sender can set $f_1 = 1$ and a best response by the adversary is to choose $p_1 = p_2 = 0$.

2.2 Computing Nash Equilibrium

Finding Nash equilibria in general non-zero games is computationally more expensive than finding equilibria in zero-sum games. In addition, there may be multiple equilibria with different payoffs for both players, which can complicate the matter of choosing a strategy. In this section we show that these concerns do not arise in the network flow game with attack costs, because the Nash equilibria in this game are precisely those of the zero-sum game where both payoffs are affected by attack cost.

To prove this we will use the following lemma:

LEMMA 1. *Let p be an adversary's strategy and let f be a sender's strategy. Then f is a best response to p if and only if f minimizes the adversary's expected payoff given p.*

PROOF. We start with the forward direction. Assume f is a best response to p. Because f is a best response to p, it follows that f must minimize harm, pMf. Therefore it must also minimize $pMf + \alpha$ for any α that is constant (with respect to f). In particular, f must minimize $pMf - pc$, the adversary's expected payoff. The proof of the reverse direction is similar. □

We can now prove the theorem:

THEOREM 1. *(f, p) is a Nash equilibrium for the network flow game with attack costs if and only if f minimizes the maximum adversary payoff and p maximizes the minimum adversary payoff.*

PROOF. We start with the backward direction. The fact that p maximizes the adversary's payoff given f follows directly from the assumption that p is a maximin strategy for the adversary's payoff. Thus p is a best response to f. It also follows that f minimizes reward given p because f is a minimax strategy for the adversary's payoff. Thus by Lemma 1 f is a best response to p. Therefore (f, p) are mutual best responses and hence form a Nash equilibrium.

Now, suppose that (f, p) is a Nash equilibrium. We prove that f must minimize the maximum adversary payoff by contradiction. Suppose that f is not a minimax strategy and let f' be a minimax strategy. Then there exists marginal probability vector p'' such that for all marginal probability vectors p', $p'Mf' - p'c < p''Mf - p''c$. In particular, for $p' = p$, we get

$$pMf' - pc < p''Mf - p''c \tag{1}$$

$$\leq pMf - pc \qquad \text{because } p \text{ is a best response to } f \tag{2}$$

But $pMf' - pc < pMf - pc$ implies that $pMf' < pMf$, which means that f is not a best response to p (for the sender), contradicting (f, p) being a Nash equilibrium. Hence f must be a minimax strategy.

It then follows readily that p must be a maximin strategy, as the attacker seeks to maximize reward. □

Because of Theorem 1, finding an equilibrium sender strategy reduces to finding a minimax strategy. This can be found efficiently by using the linear program LP 1, despite the large strategy spaces for both players:

LP 1 Equilibrium Sender Strategy with Attack Costs

Input: G, M, c, k
Output: f, R, λ

$$\underset{f,R,\lambda}{\text{Minimize}}\ kR + \sum_{a \in A} \lambda_a \tag{3}$$

subject to:

$$R \geq \text{row}_a[M]f - c_a - \lambda_a \qquad \forall a \in A \tag{4}$$

$$\sum_{(v,u)\in E} f_{vu} = b_v + \sum_{(u,v)\in E} f_{uv} \qquad \forall v \in V \backslash \{t\} \tag{5}$$

$$f_{uv} \geq 0 \qquad \forall (u, v) \in E \tag{6}$$

$$\lambda_a \geq 0 \qquad \forall a \in A \tag{7}$$

The adversary's expected payoff is represented by R and the λ_a variables. For a flow f, the *potential reward* of an attack a is the amount of additional payoff that the adversary will get if he plays a. This is calculated as $\text{row}_a[M]f - c_a$ where "$\text{row}_a[M]$" denotes the row of M corresponding to attack a. When $k = 1$, a best response by the adversary is to play an attack with maximum potential reward. Thus $\lambda_a = 0$ for all a and thus R is the amount of reward gained and will be the maximum reward that can be gained from any single attack, as required by Equation (4). Thus the sender will minimize the maximum potential reward. When $k > 1$, the sender no longer needs to minimize the potential reward of a single attack, but rather must minimize the sum of potential rewards for a set of attacks of size k. In some cases, the sender may benefit from the adversary playing attacks with higher potential reward if it allows other attacks to have lower potential reward, thus resulting in a net decrease in total potential reward. This idea is captured by the λ_a variables, which allow an attack to "borrow" potential reward from other nodes to form a net decrease. Variable R now represents the minimum potential reward among the nodes that may be attacked by the adversary in a best response. Rewriting Equation 4 to get $R + \lambda_a \geq \text{row}_a[M]f - c_a$, we see that the reward potential for a node is the minimum plus the "borrowed" amount. Each attack a will contribute $R + \lambda_a$ reward to the total, which is shown in the objective function $kR + \sum_{a \in A} \lambda_a$. LP 1 has $|A| + m + 1$ variables and at most $2|A| + n + m - 1$ constraints. Thus it can be solved in polynomial time (with respect to n and $|A|$).

For adversary, we take the dual to the sender's LP and get the following program LP 2:

LP 2 Equilibrium Adversary Strategy with Attack Costs
Input: G, M, c, k
Output: r, p

$$\underset{r,p}{\text{Maximize}} \left(\sum_{v \in S} b_v r_v \right) - p\ c^T \tag{8}$$

subject to:

$$r_u \leq r_v + p\ \text{col}_{(u,v)}[M] \qquad \forall (u,v) \in E \tag{9}$$

$$r_t = 0 \tag{10}$$

$$\sum_{a \in A} p_a \leq k \tag{11}$$

$$0 \leq p_a \leq 1 \qquad \forall a \in A \tag{12}$$

The vector p represents the marginal probabilities of attacking nodes. The vector r encodes the sender's best response to p, with r_v being the least harm that the sender can suffer for each unit of flow sent from v to the sink. At the sink, no harm can be suffered (the flow is already at the sink). From a node u other than the sink, we observe that the sender must send flow on one of the outgoing edges (u, v) to a neighbor v. The harm suffered will be equal to the harm suffered crossing (u, v), plus the harm suffered from v to t. Thus, the *least* harm suffered sending from u to t will be equal to the minimum of the harm suffered from sending flow on (u, v) plus the least harm from v to t. That is, $r_u = \min_{(u,v) \in E} r_v + p\ \text{col}_{(u,v)}[M]$ (where "$\text{col}_{(u,v)}[M]$" denotes the column in M for edge (u, v)), which is captured by the constraint in Equation 9. Because the sender needs to send b_v endogenous flow from node v to t (with $b_v = 0$ for $v \notin S$), the reward for the attacker is $(\sum_{v \in S} b_v r_v) - p\ c^T$, the objective that is maximized by LP 2.

We note that by the strong duality theorem, LP 2 finds the maximin adversary payoff strategy, despite the constraint in Equation 9 considering minimum harm (the sender's payoff), not the adversary's payoff! This phenomena is well established by Theorem 1: when the attacker optimizes his strategy against a sender who is trying to minimize harm, it is the same as optimizing the adversary strategy against a sender who is trying to minimize the adversary's payoff. That is, the sender's best response behavior in the non-zero sum game where his payoff is just based on harm and the adversary's payoff is reward is the same as the sender's best response behavior in the *zero sum* game where both players' payoffs are based on reward.

3. STACKELBERG GAME WITH ATTACK COSTS

In this section we consider the Stackelberg game in which the sender plays first, committing to a strategy. We show how two commonly used solution concepts, the strong and weak Stackelberg equilibria, are inappropriate for sequential network security games, and provide a polynomial time algorithm for finding a more nuanced equilibrium.

3.1 Model

In the previous section we described the simultaneous game where the sender and adversary act without observing each other's actions. However, in many settings this is not the case. For example, convoys in support of persistent military or humanitarian relief missions will operate over extended periods of time and the adversary can observe routes taken over time to build up an estimate of the sender's mixed strategy before choosing which attacks to launch. These types of settings are commonly modeled as *Stackelberg games*, a type of sequential game in which one player (the "leader") moves first, committing to a mixed strategy. The second player (the "follower") can then observe that mixed strategy and choose an appropriate response. It is known that in Stackelberg games the leader can sometimes improve his equilibrium payoff (and cannot decrease it, under mild assumptions) compared to his equilibrium payoff in the simultaneous move game [12].

In a two-player Stackelberg game the follower's strategy is a function that maps mixed strategies of the leader to mixed strategies of the follower. In the network flow security game with attack costs, the adversary's strategies are functions $g : \mathcal{F} \rightarrow \mathcal{A}$ that map each flow to an adversary mixed strategy. Let $\mathcal{G}$ denote the set of all such functions. A Stackelberg equilibrium is a refinement of subgame perfect Nash equilibrium where (f^*, g^*) are a Stackelberg equilibrium if they are mutual best responses, that is

$$g^*(f^*)Mf^* = \min_{f \in \mathcal{F}} g^*(f)Mf$$

$$g^*(f^*)Mf^* - g^*(f^*)c = \max_{g \in \mathcal{G}} g(f^*)Mf^* - g(f^*)c.$$

both hold, and $g^*(f)$ is a best response to f for all $f \in \mathcal{F}$ (the follower always plays optimally, even off the equilibrium path). Computing a best response function g for the adversary is straightforward: given f, greedily choose up to k attacks that have maximum payoff to the adversary, excluding any that would contribute negative payoff because the attack cost is too high. Note that there will be multiple best response functions if there is some f for which the set of k attacks yielding highest adversary payoff is not unique, and that these best response functions may yield different payoffs to the sender because of heterogeneous attack costs. Thus there may be multiple Stackelberg equilibria that have the same sender strategy but different sender payoffs.

Traditionally two kinds of Stackelberg equilibrium are distinguished: strong Stackelberg equilibrium (SSE), where the follower's best response function always maps to a strategy that maximizes the leader's payoff; and weak Stackelberg equilibrium (WSE), where the follower's best response function always maps to a strategy that minimizes the leader's payoff [8]. The pessimistic WSE is the more natural solution concept for security applications, which tend to focus on worst case behavior. Despite this, SSE, which assumes that the malicious adversary breaks ties in the leader's favor, has been considered more often in the literature for two technical reasons: (1) a SSE is guaranteed to exist in every Stackelberg game, while a WSE may not; and (2) it is often claimed that the leader can *induce* the adversary to play the desired best-case strategy by deviating by an arbitrarily

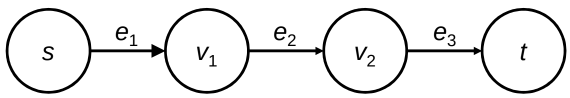

Figure 2: A network topology in which the sender cannot induce a strong Stackelberg equilibrium.

small amount from the equilibrium in order to break the adversary's indifference [12]. We will show that both of these arguments are inappropriate for the network security game, but first illustrate several important concepts by example.

Recall the example in Figure 1 with $c_1 = 100$. The adversary is indifferent when $f_1 = 103/105$, prefers the top path when it is $f_1 > 103/105$, and prefers the bottom path when $f_1 < 103/105$. Thus all best response functions $g_1 : [0, 1] \to [0, 1]$ mapping f_1 to the probability of attacking the top path must satisfy $g_1(f_1) = 0$ when $f_1 < 103/105$ and $g_1(f_1) = 1$ when $f_1 > 103/105$, and any value $g_1(f_1) \in [0, 1]$ is acceptable for $f_1 = 103/105$.

In the unique simultaneous Nash equilibrium, $p_1 = 3/105$, so that the sender was indifferent between the top and bottom paths but sent $f_1 = 103/105$ flow on the top path and $f_2 = 2/105$ flow on the bottom path, suffering harm on both paths. It follows that $f_1 = 103/105$ is a best response to the adversary's best response function g_1^{NE} with $g_1^{NE}(103/105) = 3/105$. Thus the simultaneous Nash equilibrium naturally gives rise to a Stackelberg equilibrium strategy, with the same payoff to the sender as in the Nash equilibrium, $-306/105$. In the SSE the adversary attacks the bottom path (i.e., $g_1^{SSE}(103/105) = 0$), resulting in a much higher payoff to the sender, $-6/105$. It is easy to see that there are no other Stackelberg equilibria for this game. For example, there is no WSE because if the adversary played the worst-case best response with $g_1^{worst}(103/105) = 1$, then the the sender would have incentive to deviate by decreasing f_1.

The sender's strategy is the same in both of these equilibria which means that his payoff ultimately depends on the choice of the indifferent adversary. However, note that the sender can deviate slightly from his equilibrium strategy by playing $f_1 = 103/105 - \varepsilon$ for some small $\varepsilon > 0$, in order to incentivize the adversary to attack the bottom path. By doing this the sender will receive a payoff of $-(6/105 + 3\varepsilon)$ instead of the $-6/105$ that he would earn in the SSE, but as ε is made arbitrarily small his strategy converges to the SSE strategy.

It is not always possible to induce the SSE by deviating from an equilibrium strategy. Consider the network in Figure 2, and assume that A contains two attacks, one that affects e_1 and one that affects e_2, with harm matrix

$$M = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 3 & 0 \end{bmatrix},$$

and costs $c_1 = 3$ and $c_2 = 1$. The sender has no choice as his only pure strategy is to send the full flow on the single path from s to t. At the same time, the adversary is indifferent to the choice of attack as they both yield him a payoff of 2 and so might choose either of them.

3.2 Inducing Locally Optimal Equilibria

Because the WSE may not exist and the SSE may not be attainable, we address the problem of how the sender can deviate from a Stackelberg equilibrium strategy f to induce a Stackelberg equilibrium (f, g) that yields him maximum payoff. We call this equilibrium a locally optimal inducable Stackelberg equilibrium (loptISE). It is locally optimal because the value of the Stackelberg equilibrium that is induced depend on the starting equilibrium strategy f. The starting strategy that we use is one that arises naturally from the simultaneous game Nash equilibrium strategy found by LP 1, which is always a Stackelberg equilibrium:

Algorithm 1 Computing deviation to find optimal inducable Stackelberg equilibrium

Find Nash equilibrium flow f using LP1
Set A' to be the set of minimum adversary payoff candidate attacks.
Set $k' \le k$ to be the number of candidate attacks that must be chosen from A'.
if $|A'| \le k'$ **then**
  Return.
Add dummy source s_0 to G. Set $I' \leftarrow \emptyset$. Set f^ε to be the empty flow.
while $|I| < k'$ **do**
  Set $F \leftarrow \emptyset$.
  for all $a \in A'$ **do**
    Solve $(f^a, H, H') \leftarrow \text{LP3}(G, M, A'a)$
    if $H - H' \ge 0$ **then**
      $F \leftarrow F \cup \{f^a\}$
  Set $Y \leftarrow \{a | f^a \in F$ with minimum $\text{row}_a[M]f^a\}$.
  Set $Z \leftarrow \{a | \exists a' \in Y \text{ s.t. } \text{row}_a[M]f^a = \text{row}_{a'}[M]f^{a'}\}$
  Set $A' \leftarrow A' \backslash Z$ and $I \leftarrow I \cup Z$.
  Set $f^\varepsilon \leftarrow f^\varepsilon + \sum_{a \in Y} f^a$
Set $f \leftarrow f + \varepsilon f^\varepsilon$
for all $s \in S$ **do**
  Normalize outgoing flow.

Lemma 2. *If a strategy profile (f, p) is a Nash equilibrium for the network flow security game with attack costs found by LP 1 then (f, g) is a Stackelberg equilibrium for the Stackelberg network flow security game with attack costs for a best response function g with $g(f) = p$.*

Proof. We construct g as a best response function with $g(f) = p$. For $f' \ne f$, we set $g(f')$ to be the best response that maximizes $g(f')Mf'$. By definition of Nash equilibrium, f maximizes $g(f)Mf$. Because LP 1 finds a minimax strategy, it follows that $g(f')Mf' \ge g(f)Mf$ for all $f' \in \mathcal{F}$. Thus, f is a best response to g in the Stackelberg game, and so (f, g) is a Stackelberg equilibrium. □

Algorithm 1 computes a deviation from equilibrium strategy that the sender can use to induce a loptISE. We will first sketch the high level approach before delving into the details. The algorithm starts from a Nash equilibrium flow f that will also be a Stackelberg equilibrium strategy according to Lemma 2. It then computes the set A' of *candidate attacks* for the adversary. These are the attacks that might be chosen as part of a best response to f. The sender will try to incentivize the adversary to choose certain of these candidate attacks by adding small amounts of flow to certain paths. Intuitively this approach exploits what we observed in the example: parallel paths allow the sender freedom to deviate and bias the adversary; a sequential topology does not permit that flexibility. However, the process is not as obvious when dealing with general attacks, each of which may affect an arbitrary set of links with heterogeneous penalties. Instead of choosing a simple path, the sender tries to find a flow for each candidate attack that will cause the sender to prefer to play that attack. If the attacks are "in parallel" this will be possible, if the attacks are "in sequence" it will not be. When presented with multiple options, the sender will choose the one that causes the least harm (i.e., increases his payoff the most). The process repeats until the sender has guided all of the adversary's attacks, or he has guided all the attacks that are possible. The deviation flows (which may be made arbitrarily small) are then superimposed on the original flow to generate the desired deviation.

Computing the candidate attacks is straightforward. Given f, we compute the payoff that the adversary would receive for each attack $a \in A$ as $\rho_a = \text{row}_a[M]f$. If there are k or fewer attacks with $\rho_a \geq 0$, then those are all candidate attacks. If there are more than k such attacks, we compute A' as the set of attacks with the highest k values of ρ_a (with repetition). For example, if the multiset of ρ_a values is $\{10, 10, 8, 7, 7, 7, 0, -2\}$ and $k = 4$, then $A' = \{a \in A | \rho_a \geq 7\}$, giving us 6 candidate attacks. The adversary's best response will always choose the attacks with ρ_a strictly greater than the minimum, so we need only consider A' to be those with minimum ρ_a values. In the previous example, that would mean that the best response always plays the two attacks with $\rho_a = 10$ and the one attack with $\rho_a = 8$, so we are left to choose $k' = k - 3 = 1$ candidate attacks from among the three remaining with $\rho_a = 7$.

A dummy source node s_0 is added to G and connected to each source in $s \in S$, allowing deviant flows from any source. I, the set of induced attacks, is initialized as empty. The overall deviant flow f^ε is initially empty.

The algorithm then iterates up to k' times in the loop starting at line 7. On each iteration it attempts to greedily induce the adversary to choose attacks that maximally increase the sender's payoff. F is the set of best deviant flows, computed for each candidate attack in the loop starting at line 9. The best deviant flow for a candidate attack a is computed by LP3. This LP finds a flow that causes as great as possible an increase in harm for attack a compared to any other candidate attack. For a deviation flow f^a, the adversary's best response is an attack with maximum increase in harm (attack cost does not matter as the adversary is already indifferent between candidate attacks due to the Nash equilibrium sender strategy), so if the objective value is non-negative, the adversary can be induced to play a (and possibly other candidate attacks as well). If the objective value is negative, the adversary cannot yet be induced to play a in preference to other candidate attacks.

Of the attacks that can be induced on this iteration, the sender chooses those that cause minimum increase in harm. These may not be unique, so Y is the set of all such candidate attacks with minimum increase in harm that the adversary can be induced to attack on this iteration. The deviation flows that are used to induce these attacks may also induce other attacks (which have the same increase in harm), so the set Z contains all the attacks that will be induced in the current iteration. These are removed from the candidate attacks and added to the induced attacks in line 15, and the deviation flows for this iteration are superimposed on the total deviation flow f^ε before starting a new iteration.

LP 3 Stackleberg deviating flow for a.

Input: G, M, A', a
Output: f^a, H, H'

$$\underset{f^a, H, H'}{\text{Maximize}}\ H - H' \tag{13}$$

subject to:

$$H \geq \text{row}_a[M]f^a \tag{14}$$

$$H' = \text{row}_{a'}[M]f^a \qquad \forall a' \in A' \backslash \{a\} \tag{15}$$

$$\sum_{(v,u)\in E} f^a_{vu} = \sum_{(u,v)\in E} f^a_{uv} \qquad \forall v \in V \backslash \{s_0, t\} \tag{16}$$

$$\sum_{(s_0,u)\in E} f^a_{s_0 u} = 1 \tag{17}$$

$$f^a_{uv} \geq 0 \qquad \forall (u,v) \in E \tag{18}$$

$$\tag{19}$$

The loop terminates when the requisite number of attacks have been induced. The case when no more candidate attacks can be induced is handled by there being a single flow in F which is trivially deviated from itself. For performance considerations this possibility can be checked separately. It is also not possible for A' to become empty prior to the termination of the loop. Recall also that at the beginning of iteration, $|A'| > k'$ (lines 4 – 5) and on every iteration the same number of attacks are added to I as are removed from A'. Thus, $|I| \geq k'$ no later than the iteration when $A' = \emptyset$.

In line 17 the deviation flow is scaled and superimposed on the equilibrium flow, and in line 19 the amount of flow (which increased due to the addition of the deviation flow) is normalized at each source node so that the total amount of flow is maintained with the addition of the deviation.

THEOREM 2. *Algorithm 1 runs in time polynomial in the size of G and A.*

PROOF. Each line in the algorithm can clearly be executed in polynomial time. The for loops in lines 9 – 12 and lines 18 – 19 iterate at most $\Theta(|A|)$ and $\Theta(n)$ time, respectively. In each iteration of the main loop from lines 7 – 16 at least 1 attack is added to $|I|$ and therefore the loop cannot iterate more than $|I| = \Theta(|A|)$ times. □

4. BAYESIAN GAMES WITH ATTACK COSTS

In many security applications, it is unrealistic to assume that complete information on the adversary is available because adversaries are hostile and usually secretive. In military and law enforcement domains, intelligence analysts, criminologists, and other experts collect relevant data on real and possible adversaries and develop inherently uncertain estimates of their capabilities and motives. In this section we represent that uncertainty as probability distributions over the maximum number of attacks that they can execute, the harm matrices, and the attack costs. We develop ways to reason strategically over this incomplete information by adopting the Bayesian game framework and find polynomial time algorithms for finding equilibria.

4.1 Uncertain k

In many situations it is not possible for the sender to know the adversary's capabilities with certainty. The sender can act as if he has the full knowledge, but he then might perform badly. For example, suppose that the game is the same as in Figure 1, but now $k = 2$. In equilibrium, the adversary strategy is $p_1 = 1/34$ and $p_2 = 1$, and the sender strategy is $f_1 = 100/102$ and $f_2 = 2/102$. The expected harm for the sender will thus be 3. However, if the sender does not know that $k = 2$ now, and continue to play his strategy for $k = 1$, the adversary will exploit it and will always attack v_1 and v_2. The sender's harm will thus increase to 100.116. Therefore, when the sender is not sure about the exact value of k, he will have to estimate it. We represent this by a probability distribution q over possible values of k, which we assume is known to both players. Given this distribution, we formulate the sender's problem as a *Bayesian game*. A Bayesian game is one in which information about characteristics of the other players is incomplete. There is a probability distribution over possible *types* for each player, and the type of a player determines that player's payoff function. In our case, the sender has only one type, and the type of the adversary is determined by the value of k. We denote the probability that the adversary is of type k as q_k. The sender's optimal equilibrium strategy can be computed using the following linear program:

LP 4 Equilibrium Sender Strategy in a Bayesian game (uncertain k)

Input: G, M, A, c, k, q
Output: f, $\{R^k\}$, $\{\lambda^k\}$

$$\underset{f,\{R^k\},\{\lambda^k\}}{\text{Minimize}} \sum_{k=1}^{|A|} q_k \left(kR^k + \sum_{a\in A} \lambda_a^k \right) \tag{20}$$

subject to:

$$R^k \geq \text{row}_a[M]f - c_a - \lambda_a^k \qquad \forall a \in A, \forall k \in [1..|A|] \tag{21}$$

$$\sum_{(v,u)\in E} f_{vu} = b_v + \sum_{(u,v)\in E} f_{uv} \qquad \forall v \in V \backslash \{t\} \tag{22}$$

$$f_{uv} \geq 0 \qquad \forall (u,v) \in E \tag{23}$$

$$\lambda_a^k \geq 0 \qquad \forall a \in A, \forall k \in [1..|A|] \tag{24}$$

$$R^k \geq 0 \qquad \forall k \in [1..|A|] \tag{25}$$

This LP is similar to LP 1, except that instead of minimizing maximum adversary expected payoff for a specific value of k, it minimizes the weighted sum of the expected rewards of every possible k, weighted by their possibilities. The number of variables and constraints is still polynomial in n and $|A|$ and so this LP can be solved in polynomial time.

We must verify that in the Bayesian game, minimizing the adversary's maximum expected payoff also maximizes the sender's expected payoff.

THEOREM 3. $(f, \{p^k\})$ *is a Nash equilibrium for the non-zero sum game if and only if* f *minimizes the maximum expected adversary payoff and* $\{p^k\}$ *maximizes the minimum expected adversary payoff.*

The proof is similar to the proof of Theorem 1, and we omit it due to space constraints.

Even though the adversary knows his type (i.e., the correct value of k) he cannot use LP 2 to find his equilibrium strategy since the sender does not know the exact value of k. Instead, we can take the dual to the sender's LP. Due to space constraints we omit the exact description.

5. UNCERTAIN PAYOFFS

Another way in which the sender may be uncertain of the adversary is by not knowing the payoffs and costs. Suppose instead that he has a probability distribution r over possible l payoff matrices and attack costs (types of adversaries). For $i \in [1..l]$, let r_i be the probability that the adversary is of type i with a harm matrix M^i and cost of attacks c^i. The following linear program computes the sender's optimal equilibrium strategy:

LP 5 Equilibrium Sender Strategy in a Bayesian game (uncertain M and c)

Input: G, M, c, k, r
Output: f, $\{R^i\}$, $\{\lambda^i\}$

$$\underset{f,\{R^i\},\{\lambda^i\}}{\text{Minimize}} \sum_{i=1}^{l} r_i \left(kR^i + \sum_{a\in A} \lambda_a^i \right) \tag{26}$$

subject to:

$$R^i \geq \text{row}_a[M^i]f - c_a^i - \lambda_a^i \qquad \forall a \in A, \forall i \in [1..l] \tag{27}$$

$$\sum_{(v,u)\in E} f_{vu} = b_v + \sum_{(u,v)\in E} f_{uv} \qquad \forall v \in V \backslash \{t\} \tag{28}$$

$$f_{uv} \geq 0 \qquad \forall (u,v) \in E \tag{29}$$

$$\lambda_a^i \geq 0 \qquad \forall a \in A, \forall i \in [1..l] \tag{30}$$

As before, the attacker's equilibrium strategy can be computed by taking the dual of LP 5.

If we assign $r_i = 1$ for every $i \in [1..l]$ we get a linear program which solves another interesting variant of our problem. Consider a game with one sender and multiple adversaries. The adversaries choose their strategies independently of each other (i.e., no colluding). The adversaries have different harm matrices and costs for attacking nodes and the total harm to the sender is the sum of the harm resulting from each adversary's attack. The payoff to each adversary depends only on his own strategy and the sender's strategy; it does not depend on the strategies of any of the other adversaries. For now, let's assume that every adversary can attack k nodes. By assigning $r_i = 1$ for every $i \in [1..l]$ we get that LP4 finds the optimal equilibrium strategy for the sender in the multiple adversaries game too! As for the attackers equilibrium strategies we get an interesting observation: since the strategies can be computed by the dual of LP4, they are in fact correlated. Even though the adversaries choose their strategies independently of each other, due to the strategic consideration they behave as if they coordinate their moves.

6. RELATED WORK

Problems similar to the one we address in this paper have been studied in operations research [14, 6], robotics [5, 1], and multiagent systems [13, 7]. Many of these have also taken the opposite player perspective, focusing on the player who selects nodes or edges in the network to impair the other player who chooses paths through the network. The study of network interdiction [14, 6] looks at problems where an interdictor chooses edges or nodes to destroy ("interdict") in order to impair the ability of an enemy moving through the network, for example for by forcing it to take longer paths [6]. An early study of single source, single sink zero-sum game where the interdictor could inspect a single edge found that the equilibrium strategy is to inspect only edges in the minimum cut [14]. Similar results were found in network routing settings [2], and more recently in games where multiple edges can be inpsected[13, 7]. In evader-pursuer games [5, 1], both players move through the network. The major difference with our work is in the payoff, which depends only on the probability of intercepting the path player. A major consequence of this is that there is no value for intercepting a path multiple times. In contrast, in our problem the same pathway may be attacked multiple times, thereby incurring additional harm. A version of this was recently examined in the context of communication networks [10], but this assumed zero-sum payoffs like other approaches.

A recent area of research in security games has focused on finding optimal Stackelberg strategies [11, 13] as opposed to finding the simultaneous game equilibria [2, 9]. Stackelberg games generally allow the leader to find equilibrium strategies with higher payoff than in a regular simultaneous game, but only in non-zero sum games [15, 12]. Computing the optimal strategies to commit to is solvable in polynomial time in the normal form game [3], but this is not practical in our games which have exponential sized strategy spaces.

7. CONCLUSIONS AND FUTURE WORK

In this paper we considered non-zero sum network security games where the adversary incurs costs to attack the network. We proved that the equilibria in this non-zero sum game correspond exactly to the equilibria in a related zero-sum game, and used this insight to develop linear programs (LPs) to find the equilibrium strategies. While the strategies were the same as in the zero-sum game, the payoffs were not, which allowed the sender to benefit by committing. We introduced a new Stackelberg equilibrium, the locally optimal induceable Stackelberg equilibrium (loptISE) that is particulary well suited for network security games, and provided a polynomial time algorithm for calculating the way in which the sender can deviate from an equilibrium strategy to get achieve strategy profiles arbitrarily close to the loptISE. We also found LPs to solve for equilibria in Bayesian games where the sender is uncertain of capabilities, payoffs, and costs of the adversary he faces.

In future work we seek to extend the Stackelberg framework to our Bayesian games. Commitment is computationally more difficult in Bayesian normal form games, so it will be interesting to see if we can further leverage the payoff and network structure to find polynomial time algorithms. We will also try to extend our results on loptISE to a globally optimal equilibrium, which seems possible given the relationships between the simultaneous and Stackelberg equilibria in the network flow security game.

Acknowledgments

This research has been sponsored in part by AFOSR MURI award number FA95500810356 and in part by the U.S. Army Research Laboratory and the U.K. Ministry of Defence under Agreement Number W911NF-06-3-0001.

8. REFERENCES

[1] N. Basilico, N. Gatti, and F. Amigoni. Leader follower strategies for robotic patrolling in environments with arbitrary topologies. In *AAMAS'09*, 2009.

[2] S. Bohacek, J. Hespanha, and K. Obraczka. Saddle policies for secure routing in communication networks. In *Decision and Control, 2002*, 2002.

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

[4] P. S. Efraimidis and P. G. Spirakis. Weighted random sampling with a reservoir. *Information Processing Letters*, 97(5):181 – 185, 2006.

[5] E. Halvorson, V. Conitzer, and R. Parr. Multi-step Multi-sensor Hider-Seeker Games. In *IJCAI'09*, 2009.

[6] E. Israeli and R. K. Wood. Shortest-path network interdiction. *Networks*, 40:97–111, 2002.

[7] M. Jain, D. Korzhyk, O. Vanek, V. Conitzer, M. Pechoucek, and M. Tambe. A double oracle algorithm for zero-sum security games on graphs. In *International Conference on Autonomous Agents and Multiagent Systems*, 2011.

[8] C. Kiekintveld, M. Jain, J. Tsai, J. Pita, F. Ordóñez, and M. Tambe. Computing optimal randomized resource allocations for massive security games. In *AAMAS'09*, 2009.

[9] M. Mavronicolas, V. Papadopoulou, A. Philippou, and P. Spirakis. A network game with attackers and a defender. *Operation Research*, 43(2):243–251, 1995.

[10] S. Okamoto, P. Paruchuri, Y. Wang, K. Sycara, M. Srivatsa, and J. Marecki. Multiagent communication security in adversarial settings. In *IAT'11*, 2011.

[11] P. Paruchuri, J. Pearce, J. Marecki, M. Tambe, F. Ordoñez, and S. Kraus. Playing games with security: An efficient exact algorithm for Bayesian Stackelberg games. In *AAMAS'08*, 2008.

[12] B. V. Stengel and S. Zamir. Leadership with commitment to mixed strategies. Technical report, London School of Economics, 2004.

[13] J. Tsai, Z. Yin, J. Kwak, D. Kempe, C. Kiekintveld, and M. Tambe. Urban Security: Game-Theoretic Resource Allocation in Networked Physical Domains. In *AAAI'10*, 2010.

[14] A. Washburn and K. Wood. Two-person zero-sum games for network interdiction. *Operation Research*, 43(2):243–251, 1995.

[15] Z. Yin, D. Korzhyk, C. Kiekintveld, V. Conitzer, , and M. Tambe. Stackelberg vs. nash in security games: Interchangeability, equivalence, and uniqueness. In *AAMAS'10*, 2010.