Optimal Defensive Resource Allocation for a Centrality-Based Security Game on Multi-Hop Networks

Riehl, James R.; Cao, Ming

Published in:
Proceedings of the 54th IEEE Conference on Decision and Control

2015 IEEE 54th Annual Conference on Decision and Control (CDC)
December 15-18, 2015. Osaka, Japan

Optimal Defensive Resource Allocation for a Centrality-Based Security Game on Multi-Hop Networks[1]

James R. Riehl[†] and Ming Cao[†]

***Abstract*—We present a new analysis of multi-hop network security in the form of a zero-sum game played between an attacker who tries to disrupt a network by disabling one or more nodes, and the nodes of the network who must allocate limited resources in defense of the network. The payoffs in the zero-sum game can be one of several performance metrics that correspond to node centrality measures. In the case of single-node attacks, we use a monotonicity property of the mixed attack strategies to construct a simple and very fast algorithm to compute saddle-point equilibrium strategies for both single-node and multiple-node defense. For simultaneous multiple-node attacks on large networks, the computational complexity becomes quite high, so we present a method to approximate the equilibrium strategies based on a sequential simplification, which performs well in simulations.**

I. Introduction

As society grows ever more reliant on complex and dynamic networked systems for applications such as communication, transportation, distributed sensing, and control, these systems are increasingly vulnerable to attacks by malicious adversaries, and it becomes critical to secure them against such attacks. However, both attack and defense to a network generally require significant energy or resources, and there may not be enough to cover an entire network. For example, a distributed denial of service (DDoS) attack requires energy and bandwidth from multiple coordinating attackers who try to deplete the memory or bandwidth resources of the targeted system. Similarly, such an attack can be defended by adding memory or bandwidth capacity to vulnerable nodes in the network [1]. The decision on how to allocate these resources to the nodes is therefore crucial to the effective defense of the network. In this paper, we consider this problem from the perspective of a central network administrator who has full information about the network and is responsible for maximizing one of several possible performance metrics that can be mapped to node centrality measures.

Although centrality measures were originally introduced in the context of social network analysis, their use has already gained traction in the communication and control literature. In particular, the authors of [2] studied the effects of coordinated attacks on wireless mesh networks and showed that targeting the nodes with the highest betweenness centrality results in a more effective attack than targeting based on degree, a result supported by earlier studies on the attack vulnerability of complex networks [3]. However, this strategy might easily be predicted and thwarted by a smart defender. Game theory offers a set of tools that are ideally suited to such competitive settings and for this reason has become widely adopted in the study of network security [4][5]. Despite these emerging research trends, this work is the first we are aware of to combine the tools of social network analysis and game theory towards the solution of a highly practical network security problem.

The problem we consider here is essentially one of resource allocation in the presence of an adversary, and there does exist some relevant literature on this topic. For example in [6], two teams allocate power between communicating with teammates and jamming the other team's communications. After formulating the problem as a zero-sum differential game, sufficient conditions on the agent parameters are given for existence of a pure-strategy Nash equilibrium. The optimal design of resource allocation networks subject to attacks is studied in [7], where it is shown that networks with a star topology are optimal from the perspective of a defender. In a transportation setting, path-planning against adversaries was formulated as a distributed resource allocation problem and a linear-programming solution was proposed [8]. Finally, agent-based simulations are used in [9] to determine Nash equilibrium strategies in a network security game where both attacker and defender are subject to cost constraints.

In this paper we introduce a linear-time algorithm to compute saddle-point equilibrium strategies for both the attacker and defender in the case of single-node attacks and multiple-node defense. When multiple nodes are attacked simultaneously, we propose an approximation algorithm based on a sequential simplification of the attack strategies. These algorithms provide a network administrator with the probabilities that each node should be protected. An extension that allows distributed computation of the defense policies is considered in [10].

[1]The work was supported in part by the European Research Council (ERCStG-307207).

[†]Faculty of Mathematics and Natural Sciences, ENTEG, University of Groningen, The Netherlands, {j.r.riehl, m.cao}@rug.nl

II. Preliminaries

Before we address the central question of this paper, we briefly review two subjects that are fundamental to understanding the problem and solution approach: node centrality and zero-sum game theory.

A. Node Centrality

The *centrality* of a node can be thought of as the importance of a node in the context of the network, and there are many different measures of centrality related to various notions of importance. One notable example is *betweenness*, which is the fraction of all shortest paths in the network on which a node lies. Consider the five-

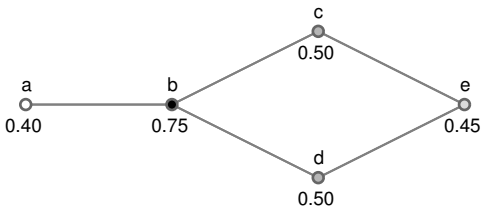

Fig. 1. Betweenness centrality on a small network

node undirected network shown in Fig. 1. Their are ten different node pairings in this network but twelve paths that are shortest paths between pairs of nodes:

$$ab, abc, abd, \{abce, abde\}, bc, bd, \\ \{bce, bde\}, \{cbd, ced\}, ce, de, \quad (1)$$

where bracketed sets of paths indicate when there are multiple shortest paths connecting two nodes. The betweenness of each node is then obtained by counting the total number of shortest paths in which a node appears, dividing by k when the path is one of k shortest paths connecting two nodes. Finally, we divide by the total number of node pairs in the network so that the result lies between zero and one. Checking each path in (1), we see that node b lies on $1+1+1+(\frac{1}{2}+\frac{1}{2})+1+1+(\frac{1}{2}+\frac{1}{2})+\frac{1}{2} = 7.5$ out of the 10 node pairs, or 0.75 of the shortest paths in the network, supporting the intuitive observation that b is an important node in this communication network.[†] Although we focus on the example of betweenness in this paper, the results apply to any network performance metric that can be decomposed into individual node contributions via a centrality measure.

[†]Our definition of betweenness actually differs slightly from standard betweenness definitions because we count being an endpoint of a path as being on the path while standard definitions typically do not. The reason is that for our purposes the endpoint nodes are just as important to the communication link as any intermediate node.

When multiple nodes are attacked simultaneously, it will not generally be possible to measure the effect of the attack on network performance from the individual node centrality values. Therefore we need to define the centrality of a *node set*, which we can motivate with an example. Consider two-node attacks to sets $\{a, b\}$ and $\{b, e\}$ in the network of Fig. 1. We want to evaluate the removal of these node pairs from the network, which in the case of betweenness, corresponds to the fraction of shortest paths containing at least one of the removed nodes. For $\{a, b\}$ this comes to 0.75 since every path that contains a also contains b. On the other hand, node e is on every shortest path that does not contain b, so the betweenness of $\{b, e\}$ is 1. Similar modifications can often be made to adapt other single-node centrality measures to apply to node sets.

B. Zero-Sum Game Theory

In a *game*, two or more players choose from a set of strategies in pursuit of different and often competing objectives and receive payoffs that depend on the strategies of all players. A *two-player zero-sum game* is a game in which one player's gain is equal to the other player's loss and is commonly modeled using a payoff matrix where one player is the minimizer and the other is the maximizer. The players choose from a set $\mathcal{S}$ of finite pure strategies, but they can randomize their choices in what is called a mixed-strategy, *i.e* a vector of length $|\mathcal{S}|$ whose elements are all non-negative and sum to one. Two-player games with finite strategies always admit a unique saddle-point equilibrium in mixed strategies [11]. A well-known example of this is in the rock-scissors-paper game where the equilibrium corresponds to both players using each of {rock, scissors, paper} with probability one-third. This result illustrates a key concept in game theory, which is the *principle of indifference*, the idea being that if I am playing a saddle-point equilibrium mixed strategy, my opponent can pick from several strategies without seeing any change in payoff, and is thus indifferent to which strategy to play.

III. Problem Setup

In this formulation, we assume that attacks target one or more nodes in the network, and that an attack on an unprotected node results in the removal of that node from the network. On the other hand, a protected node is immune to attack and cannot be removed. Taking the viewpoint of a defender who wants to maximize the performance of the network in the presence of worst-case attacks, we assume that the attacker has full information about the network except for which nodes are defended.

Since a worst-case attacker wants to minimize the same performance metric that the defender wants to maximize, this problem fits well into the setting of a two-player zero-sum game, where the attacker tries to maximally disrupt the network by disabling one or more nodes and a defender must choose which nodes to protect in order to best defend the network against such an attack. We can express the problem in general form as the following minimax problem:

$$v^* = \min_{y \in \Delta_{\mathcal{A}}} \max_{z \in \Delta_{\mathcal{D}}} y^T A z, \tag{2}$$

where y and z are mixed strategies of the attacker and defender, respectively, and v^* is the expected performance change to the network. We denote by $\Delta_{\mathcal{A}}$ and $\Delta_{\mathcal{D}}$ the mixed-strategy simplexes over pure strategy attack and defense sets $\mathcal{A}$ and $\mathcal{D}$, which will depend on the number of nodes attacked (α) and defended (δ).

A. Single-Node Attack / Single-Node Defense

We begin with the case where the attacker and defender each have only enough resources to attack or defend one node, resulting in pure strategy sets that map directly to the nodes in the network, *i.e.* $\mathcal{A} = \mathcal{D} := \mathcal{V}$, where $\mathcal{V} := \{1, \ldots, n\}$. To construct the payoff matrix, we observe that each entry A_{ij} should correspond to the change in network performance when node i is attacked and node j is defended. Since a defended attack results in no change to the network, A should have zeros on the diagonal. All other entries correspond to an undefended attack on node i, whose removal from the network results in a performance loss exactly equal to its centrality value, which we denote by a_i. The payoff matrix can therefore be expressed as

$$A = \begin{array}{c} \\ att_1 \\ att_2 \\ \vdots \\ att_n \end{array} \begin{array}{c} \begin{array}{cccc} def_1 & def_2 & \cdots & def_n \end{array} \\ \begin{pmatrix} 0 & -a_1 & \cdots & -a_1 \\ -a_2 & 0 & \cdots & -a_2 \\ \vdots & \vdots & \ddots & \vdots \\ -a_n & -a_n & \cdots & 0 \end{pmatrix} \end{array}, \tag{3}$$

where att_i and def_j respectively indicate attack to node i and defense of node j.

B. Single-Node Attack / Multiple-Node Defense

In the previous section we assumed that both the attacker and defender could only target a single node, but it is easy to imagine that the defender might have enough resources to protect $\delta > 1$ nodes. In this case, the attack strategy set $\Delta_{\mathcal{A}}$ remains the same as before, but the defense strategy set expands to cover all possible combinations of defended nodes. Let $\mathcal{V}_k$ denote the set of all subsets consisting of k distinct nodes in the network. Then we have the pure strategy set $\mathcal{D} := \mathcal{V}_\delta$ and $\Delta_{\mathcal{D}}$ becomes the mixed-strategy simplex over $\mathcal{D}$. Let $\mathcal{D}(j)$ denote the j^{th} set of δ nodes under some enumeration of the set $\mathcal{D}$. Each entry A_{ij} in the payoff matrix now corresponds to the expected change in network performance when node i is attacked and the nodes $\mathcal{D}(j)$ are defended. Therefore $A_{ij} = 0$ whenever $i \in \mathcal{D}(j)$ and $A_{ij} = -a_i$ otherwise.

C. Multiple-Node Attack / Multiple-Node Defense

Next we consider the case where the attacker also has enough resources to target $\alpha > 1$ nodes. Now the pure attack strategy set is $\mathcal{A} := \mathcal{V}_\alpha$, the pure defense strategy set remains $\mathcal{D} := \mathcal{V}_\delta$, and each entry A_{ij} corresponds to the expected network performance change when nodes $\mathcal{A}(i)$ are attacked and nodes $\mathcal{D}(j)$ are defended. Recall that we defined the centrality of a node set in Section II-A as the effect on network performance when that set of nodes is removed from the network. Hence A_{ij} is equal to the centrality of the node set $\mathcal{A}_\alpha(i) - \mathcal{D}_\delta(j)$ whenever this set is nonempty, and 0 otherwise.

IV. Solution: Single-Node Attacks

Problem (2) is in a standard form that is known to be solvable by linear programming (LP), and although there are LP algorithms that run in polynomial time [12], this can still be a computational burden when working with very large networks. Moreover, the payoff matrices grow combinatorially with the number of attack and defense nodes, providing an even stronger motivation to develop faster solution algorithms. In this section we present a simple and very fast algorithm to compute the saddle-point equilibrium strategies against single-node attacks, made possible by the following facts that follow from standard properties of mixed-strategy saddle-point equilibria (proofs can be found in [10]):

1) Equilibrium mixed-strategies correspond to a subset of nodes, which we call *support* nodes
2) Support nodes are those with the highest centrality
3) Principle of indifference applies to support nodes
4) At equilibrium, all non-support pure strategies are strictly dominated

The key insight behind our approach is that the expected payoff resulting from a mixed attack strategy is monotone in the addition of pure strategies or nodes. In other words, when choosing between two nodes to add to a mixed attack strategy, it is always better to choose the node with the higher centrality. This means

we can incrementally construct a mixed defense strategy by adding support nodes in order of decreasing centrality until all remaining pure strategies are dominated.

We start by sorting the node centralities from highest to lowest in the vector $\tilde{a} := Pa$, where P is the appropriate permutation matrix. Algorithm 1 then proceeds as follows. The counting index k is initialized to 1 and incremented for every node that is added to the mixed strategy, while σ_k is the negated cumulative sum of reciprocal centralities. These quantities are used to update the expected payoff value v_k, and the loop is repeated until either v_k is less than the next highest negated centrality value $-\tilde{a}_{k+1}$ or $k = n$. Note that if $v_k < -\tilde{a}_{k+1}$, that means all remaining pure strategies are strictly dominated, *i.e.* there is no reason for the attacker to consider attacking any of the remaining nodes because it would only decrease the effectiveness of the attack. This value of v_k is in fact the saddle-point equilibrium value, and the corresponding strategies $\tilde{y}$ and $\tilde{z}$ are calculated in steps 9 and 10.

1 $k := 1$
2 $\sigma_k := \frac{1}{-\tilde{a}_1}$
3 **repeat**
4 $\quad k := k + 1$
5 $\quad \sigma_k := \sigma_{k-1} + \frac{1}{-\tilde{a}_k}$
6 $\quad v_k := \frac{k-\delta}{\sigma_k}$
7 **until** $v_k < -a_{k+1}$ *or* $k = n$
8 **foreach** $i \in \{1, \ldots, k\}$ **do**
9 $\quad \tilde{y}_i := \frac{1}{-\tilde{a}_i \sigma_k}$
10 $\quad \tilde{z}_i := 1 - \frac{v_k}{-\tilde{a}_i}$
11 **end**
12 $v^* := v_k$

Algorithm 1: Computes saddle-point equilibrium value and strategies of (2) for single-node attacks.

The equilibrium strategies for the original game can then easily be constructed by using the inverse ordering of $\tilde{a}$ to assign the k values of $\tilde{y}$ and $\tilde{z}$ to the corresponding entries in y and z and setting the remaining values to zero. The following theorem confirms that this algorithm achieves the desired result for single-node attack and defense (proof in [10]).

Theorem 1: Algorithm 1 computes the unique saddle-point equilibrium value v^* to problem (2) for single-node attack and single-node defense strategies.

It turns out that Algorithm 1 can also be used to solve for the case of multiple-node defense with only a small modification. To see how, note that the particular combination of nodes defended does not matter in the single-node attack case. All that does matter is whether a node is defended or not. Recall that the mixed strategy vector z now has length $\binom{n}{\delta}$ corresponding to the pure strategies of defending every possible combination of δ nodes, but let us define a new n-length vector $\bar{z}$, which contains the probability that each node in the network is defended. We will also define $\tilde{z} = P\bar{z}$. Note that $\bar{z}$ is *not* a mixed strategy since its values do not sum to one, but it serves the purpose of the algorithm anyway, and later we can construct a mixed-strategy z by solving the underdetermined system of equations $Az = diag(-a)(\mathbf{1}_n - \bar{z})$. The following theorem extends Theorem 1 to the case of multiple-node defense (proof in [10]).

Theorem 2: Algorithm 1 computes the unique saddle-point equilibrium value v^* to problem (2) for single-node attack and multiple-node defense strategies.

***Remark** 1:* The inside of each loop in Algorithm 1 contains a fixed number of operations and is repeated a maximum of n times resulting in a total number of operations that is in the worst case $O(n)$, *i.e.* linear in the number of nodes in the network. For comparison purposes, polynomial-time algorithms for linear programming have computational complexity in the neighborhood of $O(n^{3.5})$ [13].

V. Approximation: Multiple-node attacks

The possibility of simultaneous attacks to multiple nodes adds additional complexity to the problem because the specific combinations of nodes attacked and defended becomes more important. We can no longer consider the removal of nodes from the network in isolation, but rather we must capture the inter-dependencies of nodes on network performance through the concept of node-set centrality, introduced in Section II-A. Although the resulting problem (2) remains computable by linear programming, the size of the payoff matrix grows combinatorially in the number of nodes attacked and defended. As an example, the payoff matrix for 5-node attack and defense strategies on a 50 node network contains over 2 million $\times$ 2 million entries. Unfortunately, we no longer have monotonicity of the mixed-strategies and therefore cannot use Algorithm 1 to simplify and solve this problem exactly.

However, there is a technique that can greatly reduce the computation at a minimal cost to optimality. The key is to realize that similar to the single-node attack case, it is likely that in most practical networks only a few of the large number of possible attack combinations will play a role in the equilibrium solution. If we could somehow identify the most effective attacks without computing the entire payoff matrix, we could save a substantial amount of computation. The method we propose in the

next section computes the most effective *sequences* of attacks in order to predict the most effective *parallel* attack strategies from the full payoff matrix.

A. Sequential Approximation Algorithm

Algorithm 2 begins by using Algorithm 1 to find the support nodes for a single-node attack (step 1) and adds these nodes to the initial attack set (step 2) which we will augment incrementally (step 3) using the following procedure. Starting from an initially empty set of new attack nodes (step 4), for each attack combination in the previous iteration (step 5), we remove those nodes from the network (step 6) and update the centrality values for the remaining nodes on the reduced network $\hat{\mathcal{V}}$ (step 7), resulting in the new payoff matrix $\hat{A}$. We then reuse Algorithm 1 in sequence to find the next set of support nodes (step 8) (presuming the attacker has successfully removed the nodes $\hat{\mathcal{A}}^{k-1}(i)$ from the network). This step is the essence of the sequential simplification. Next we construct new attack sets from every unique combination of the $k-1$-node attack sets and the new support nodes (step 9) and then repeat the process. Finally, we set the defense set equal to the attack set, adjusting for size if $\delta \neq \alpha$ (step 12) and solve the reduced game using linear programming (step 13).

1 $\hat{y} := \arg\min_{y \in \Delta_{\mathcal{V}}} \max_{z \in \Delta_{\mathcal{V}_{\delta+1}}} y^T A z$
2 $\hat{\mathcal{A}}^1 := \{i : \hat{y}_i > 0\}$
3 **for** $k := 2$ *to* α **do**
4 $\hat{\mathcal{A}}^k := \emptyset$
5 **for** $i := 1$ *to* $|\hat{\mathcal{A}}|$ **do**
6 $\hat{\mathcal{V}} := \mathcal{V} - \hat{\mathcal{A}}^{k-1}(i)$
7 Compute centralities $\hat{a}$ on $\hat{\mathcal{V}}$
8 $\hat{y} := \arg\min_{y \in \Delta_{\hat{\mathcal{V}}}} \max_{z \in \Delta_{\hat{\mathcal{V}}_{\delta-k+2}}} y^T \hat{A} z$
9 $\hat{\mathcal{A}}^k := \hat{\mathcal{A}}^k \cup \hat{\mathcal{A}}^{k-1}(i) \times \{j : \hat{y}_j > 0\}$
10 **end**
11 **end**
12 $\hat{\mathcal{A}}^k \to \hat{\mathcal{D}}$
13 $\hat{v} := \min_{y \in \Delta_{\hat{\mathcal{A}}^\alpha}} \max_{z \in \Delta_{\hat{\mathcal{D}}}} y^T A z$

Algorithm 2: Algorithm to approximate saddle-point equilibrium value and strategies of (2) for the case of multiple-node attacks based on a sequential simplification.

B. Simulations

It is quite difficult to derive analytical measures for how closely Algorithm 2 approximates the saddle-point equilibrium defense strategy in general, but we can test it on a model of a real network as well as a range of randomly generated networks and compare to other potential defense strategies, including the exact solution of (2) provided the networks are small enough. We begin by testing the approach on connectivity data of the UCSB MeshNet, a real experimental wireless mesh network dataset that was also used in [2].

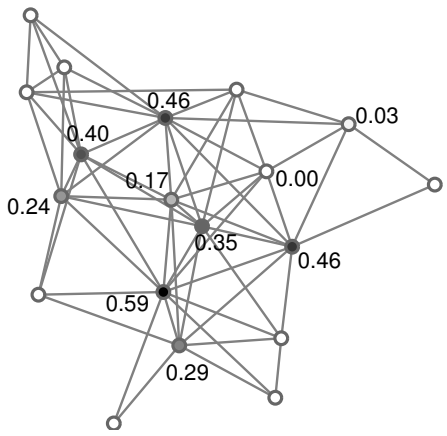

Fig. 2. Results of Algorithm 2 applied to the UCSB MeshNet data from a particular instant of time. The shading of the nodes is proportional to the adjacent numbers which are the probabilities that each node will be defended as part of a three-node mixed defense strategy against a three-node attack.

Fig. 2 shows the network along with the computed mixed defense strategy for three-node defense against a three-node attack. We compared the sequential approximation to two other defense strategies: (1) a *pure strategy*, which defends the three nodes having the highest betweenness centrality, and (2) *random sampling*, which samples the payoff matrix by randomly choosing 20% of the possible node combinations and solving the game on the resulting reduced payoff matrix. The random sampling approach is inspired by [14] and has the advantage of approximation guarantees assuming the attacker is also using random sampling. However, in order to make an unbiased comparison between strategies, we assume the attacker plays the equilibrium strategy against whatever strategy the defender is using. For the pure strategy, this means the attacker knows the defender will defend the three most central nodes and thus will not attack those nodes, while for the sequential approximation, this means the attacker knows which columns of the payoff matrix the defender has chosen. Table I shows the expected payoffs for each of these defense strategies as well as the true saddle-point equilibrium strategy. Notice that the attacker playing against the sequential approximation can still not improve on the equilibrium payoff despite having a significant advantage.

TABLE I
MULTIPLE-NODE ATTACKS TO UCSB MESHNET

Defense strategy	Nodes attacked/defended		
	1	2	3
Saddle-Point Equil.	-0.171	-0.273	-0.342
Sequential Approx.	-0.171	-0.273	-0.342
Random Sampling	-0.193	-0.279	-0.349
Pure Strategy	-0.232	-0.402	-0.539

Next, we tested the accuracy of Algorithm 2 against the two other approaches for a two-node attack and defense scenario on 100 random geometric networks of increasing size, generated by randomly placing n nodes the unit square and connecting any pair of nodes that were closer than a distance threshold of 0.4 from each other. Fig. 3 shows the expected fraction of performance loss ($\frac{\hat{v}-v^*}{v^*}$) of the three approaches compared to the equilibrium payoff, and Fig. 4 compares the average time required to compute the sequential approximation to that for the full payoff matrix and linear programming (LP) solution. We see that the sequential approximation comes quite close to the exact solution with a relatively flat computation time compared to the LP.

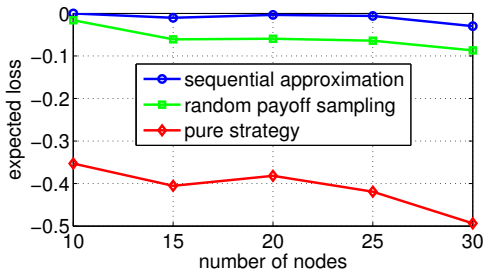

Fig. 3. Mean fraction of expected performance loss of the three approximation methods over 100 random geometric networks of five different sizes.

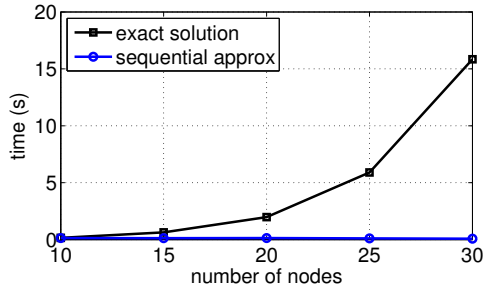

Fig. 4. Mean computation times of the sequential approximation compared to solving the exact solution via linear programming.

VI. CONCLUSIONS AND FUTURE WORK

In this paper we have formulated a topological network security problem as a zero-sum game between an attacker and defender with the goal of determining which nodes to protect in a network when resources are limited. In the case of a single-node attack, we presented a simple and fast algorithm to compute the equilibrium strategies for both single-node and multiple-node defense. For multiple-node attacks the problem increases in complexity, so we investigated a sequential approximation to the saddle-point equilibrium defense strategy and simulations showed that this method closely approximates the exact solution at greatly reduced computation. We are currently working on extending the process to the distributed setting, using only local information to compute defense strategies [10].

REFERENCES

[1] S. Arunmozhi and Y. Venkataramani, "A new defense scheme against DDoS attack in mobile ad hoc networks," in *Advanced Computing*. Springer, 2011, pp. 210–216.

[2] M. Kas, S. Appala, C. Wang, K. M. Carley, L. R. Carley, and O. K. Tonguz, "What if wireless routers were social?" *Wireless Communications, IEEE*, vol. 19, no. 6, pp. 36–43, 2012.

[3] P. Holme, B. J. Kim, C. N. Yoon, and S. K. Han, "Attack vulnerability of complex networks," *Physical Review E*, vol. 65, no. 5, p. 056109, 2002.

[4] Z. Han, D. Niyato, W. Saad, and A. Hjørungnes, *Game theory in wireless and communication networks: theory, models, and applications*. Cambridge University Press, 2011.

[5] S. Roy, C. Ellis, S. Shiva, D. Dasgupta, V. Shandilya, and Q. Wu, "A survey of game theory as applied to network security," in *System Sciences (HICSS), 2010 43rd Hawaii International Conference on*. IEEE, 2010, pp. 1–10.

[6] S. Bhattacharya, A. Khanafer, and T. Başar, "Power allocation in team jamming games in wireless ad hoc networks," in *Proceedings of the 5th International ICST Conference on Performance Evaluation Methodologies and Tools*. ICST, 2011, pp. 515–524.

[7] S. Goyal and A. Vigier, "Attack, defence, and contagion in networks," *The Review of Economic Studies*, vol. 81, no. 4, pp. 1518–1542, 2014.

[8] G. C. Chasparis and J. S. Shamma, "Linear-programming-based multi-vehicle path planning with adversaries," in *American Control Conference, 2005. Proceedings of the 2005*. IEEE, 2005, pp. 1072–1077.

[9] A. Nochenson and C. L. Heimann, "Simulation and game-theoretic analysis of an attacker-defender game," in *Decision and Game Theory for Security*. Springer, 2012, pp. 138–151.

[10] J. Riehl and M. Cao, "A centrality-based security game for multi-hop networks," *Submitted for journal publication*, 2015.

[11] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory, 2nd Edition*. SIAM, 1999.

[12] N. Karmarkar, "A new polynomial-time algorithm for linear programming," in *Proceedings of the sixteenth annual ACM symposium on Theory of computing*. ACM, 1984, pp. 302–311.

[13] G. Strang, "Karmarkars algorithm and its place in applied mathematics," *The Mathematical Intelligencer*, vol. 9, no. 2, pp. 4–10, 1987.

[14] S. D. Bopardikar, A. Borri, J. P. Hespanha, M. Prandini, and M. D. Di Benedetto, "Randomized sampling for large zero-sum games," *Automatica*, 2013.