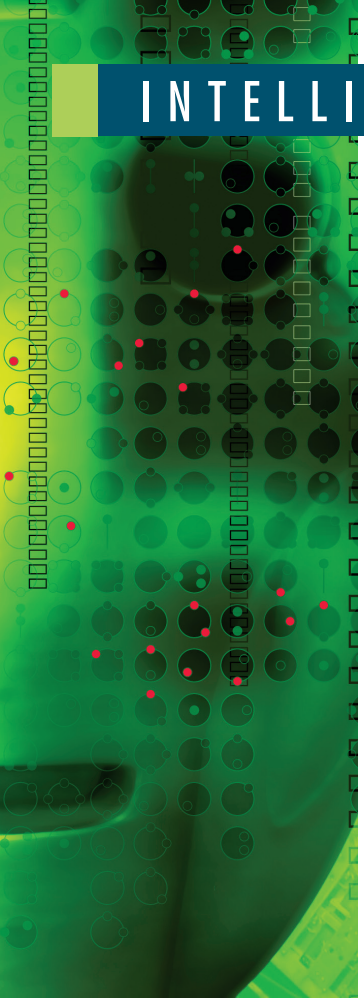

# Case Studies of Network Defense with Attack Graph Games

**Karel Durkota,** *Czech Technical University*

**Viliam Lisý,** *Czech Technical University and University of Alberta*

**Christopher Kiekintveld,** *University of Texas at El Paso*

**Branislav Bošanský and Michal Pěchouček,** *Czech Technical University*

*A challenge many existing tools fail to address is that attackers react strategically to new security measures, adapting their behaviors in response. Game theory provides a methodology for making decisions that take into account these reactions.*

Computer network security is an example of asymmetric, strategic conflict between defenders and attackers, with attackers performing a wide range of intrusion actions such as scanning the network and exploiting vulnerabilities, and defenders countering with actions such as intrusion detection and filtering. Different defense mechanisms have varying costs and effectiveness against specific types of attacks, forcing network administrators to make challenging decisions about how best to optimize the ways in which they select and deploy these measures. Initial effectiveness can be mitigated in the long term as attackers adapt to security measures, making optimization even more challenging.

Game theory provides a methodology for developing new decision support tools that takes into account attackers' sophisticated responses to defense strategies. The key idea is that rational attackers will respond to security, so we can model the network defense problem as a two-player, multistage game. Solving these models lets us find an optimal defense plan to mitigate attacks and can also provide a quantitative measure of security improvement. We demonstrate this methodology for one specific type of defensive strategy: honeypots, which are fake hosts or services added to a network. Honeypots act as decoys to distract attackers from real hosts, detect the presence of intruders, and gather detailed information about an attacker's activity.

Operating believable honeypots is expensive in terms of hardware, software, and administrator time (typically spent managing the honeypot and analyzing data). In addition, many different types of honeypots could be used in any given network. We describe a game-theoretic approach for optimizing honeypot deployments as a case study for how game theory can be used to make network security decisions. Our case study on a realistic network shows the feasibility of this methodology.



1541-1672/16/$33.00 © 2016 IEEE
*Published by the IEEE Computer Society*

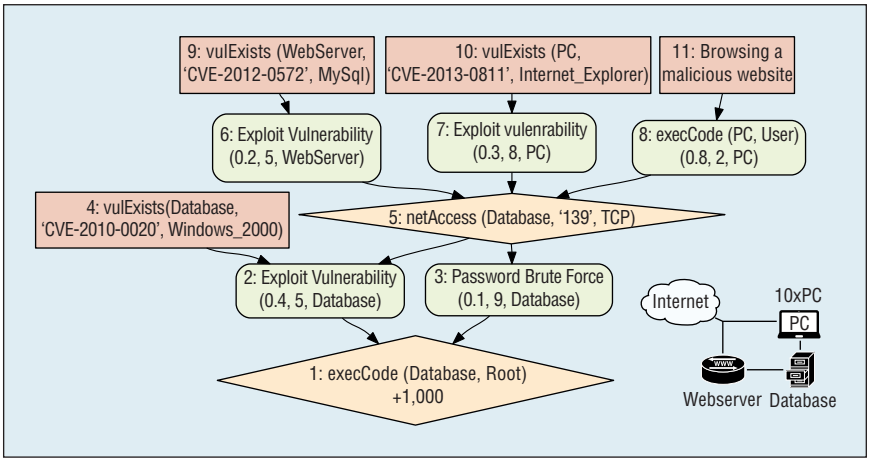


**Figure 1. A network and an attack graph representing ways to gain access to the database. Each action (rounded rectangular node) has preconditions and effects, represented by incoming and outgoing edges, respectively. Initially true facts are represented with rectangle nodes and initially false facts with diamond nodes.**

## Background

Network administrators and security researchers use honeypots to detect and analyze attacker behaviors and tools.[1] In particular, the Honeynet Project provides a wealth of software and literature on knowledge acquired about attackers. Companies use honeypots as intrusion detection systems (IDSs)—hardware products (for example, Canary by Thinkst) that can emulate various OSs. To effectively use honeypots, network administrators must decide how many to deploy (contingent on cost) and where to place them to make them attractive targets.

### Game Theory

Game theory models decision-making problems with multiple decision makers (players) in a common, often partially observable, environment. A game consists of players, strategies (actions) available to each player, and utilities for each defender depending on the joint choices of all players; players can also have incomplete information about moves made by other players or the environment. The optimal strategy for a player generally depends on other players' behavior.

Game theory provides a variety of solution concepts and algorithms for analyzing games with different characteristics. We focus on two-player games where the administrator (defender) chooses a honeypot allocation that minimizes the cost and expected loss caused by an attacker compromising the network. The attacker chooses a strategy that maximizes the value of attacking the network while avoiding honeypots and minimizing costs. We use Stackelberg game models similar to those used for physical security domains.[2] The defender acts first, taking actions to harden the network by adding honeypots. The attacker then chooses the optimal attack plan based on (limited) knowledge about the network and the defender's strategy. Solving the game means computing a strategy for each player, which describes an optimal (stochastic) action choice in every possible situation.

### Attack Graph

Attack graphs (AGs) represent possible sequential attacker strategies for compromising a specific computer network. They're automatically generated based on known vulnerabilities[3] and are used to identify the minimal subset of vulnerabilities to be patched or sensors to be placed in the network to prevent known attacks, to calculate security risk measures,[4] or to find the shortest attack plan in penetration testing. We use AGs to compactly represent possible attack plans (attacker strategies) in our game models.

## Model Overview

We model a network as a set of host types, such as the webserver or PC in Figure 1. Two hosts are of the same host type if they run the same services and have the same network connectivity and value. For example, a collection of workstations that run the same OS image are modeled as the same type. The host-type PC in Figure 1 has 10 equivalent hosts, all of which present the same attack opportunities. By representing each type only once in the attack graph, we can scale with the number of unique types rather than of individual hosts.

### Defending

Let's say the defender places $k$ honeypots in the network by duplicating the configurations of existing hosts (with obfuscated data) or creating new host types. The defender pays a cost that's dependent on the host type for each honeypot, so adding more honeypots of a specific host type increases the likelihood that the attacker will interact with a honeypot instead of a real host. If the attacker interacts with a honeypot during an attack, an alert is sent to the defender, who immediately stops the attack or applies other countermeasures.

### Attacking

In our model, we consider exploit actions that target a specific vulnerability in a host. Successfully executing an exploit results in the attacker gaining privileged or nonprivileged access to that host.

An AG compactly represents all known sequences of exploit actions for the host in the network. Specifically, it captures the dependencies between the exploit actions and true or

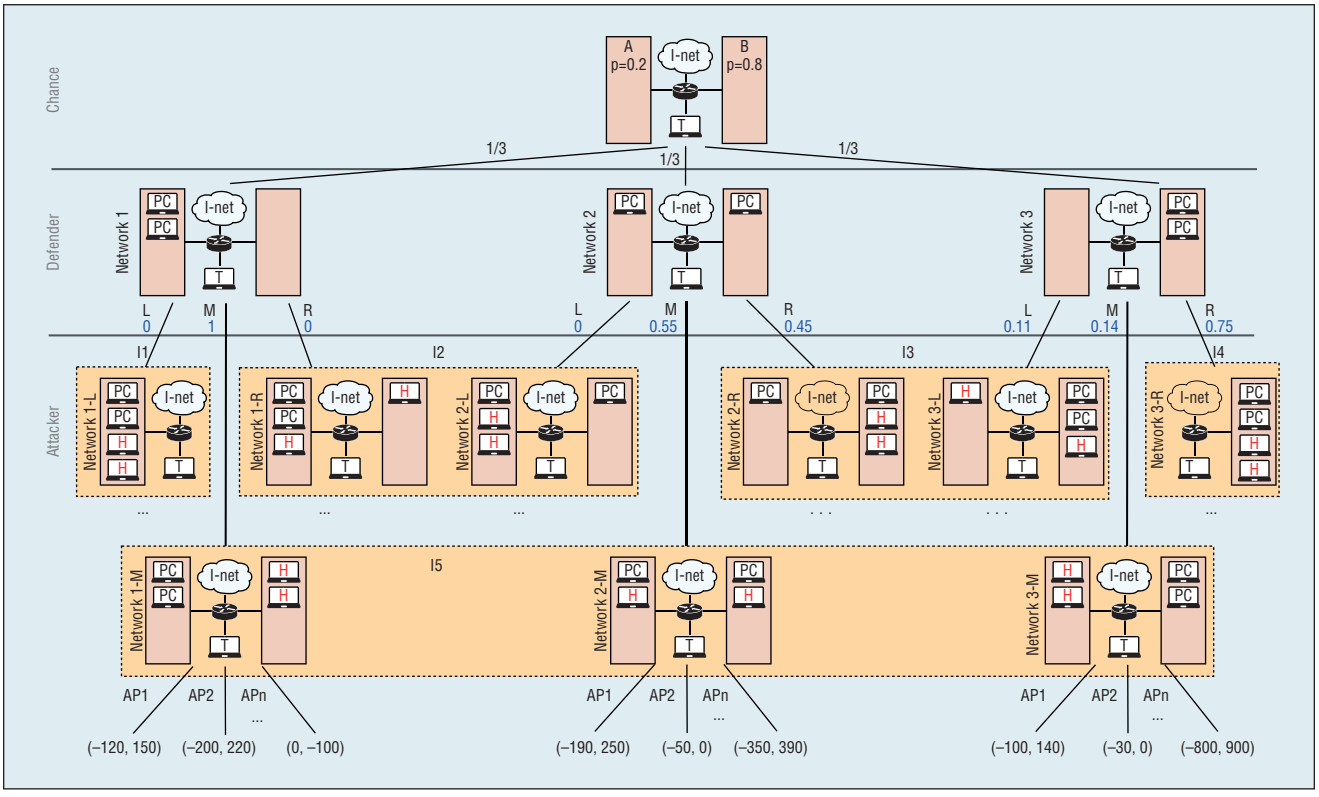


**Figure 2. Game tree for a simple network of host types A and B, with attack success probabilities 0.2 and 0.8, respectively. Chance plays uniformly into three possible networks, each containing two hosts. The defender chooses possible combinations of allocating two honeypots in each possible network; the blue probability value is a possible defense strategy. The attacker selects attack policy $AP_1$ through $AP_n$ according to the networks' attack graphs.**

false facts that represent logical statement about the network state. Figure 1 depicts a possible AG for the network; each action (rounded rectangular node) has preconditions and effects, represented by incoming and outgoing edges, respectively. All preconditions must be true to perform the action. If an exploit action succeeds, then all its effects become true and the attacker obtains rewards. Initially true facts are represented with rectangle nodes and initially false facts with diamond nodes.

In Figure 1, action "`2:Exploit Vulnerability`" has preconditions: Fact 4, vulnerability exists, and Fact 5, the attacker can access the daase on port 139. If the attacker successfully performs the action, he will obtain root privileges to the database (Fact 1) and reward +1,000. The success probability is the likelihood that an exploit will succeed, given that all preconditions are met, and the cost represents the attacker's monetary cost and effort for attempting to perform an action, as well as the consequences of possibly disclosing the exploit. Action 2 in Figure 1 has a success probability of 0.4 and a cost of 5.

We use MulVAL[3] to automatically construct AGs from information collected by network scanning tools (such as Nessus or OpenVAS). The action costs and success probabilities can be estimated using the Common Vulnerability Scoring System[5] or other data sources.

In our game models, the attacker chooses the optimal attack policy that fully characterizes the attacker's behavior at every point in an attack—it specifies the order of the actions to be executed by a rational attacker. The problem of computing the optimal attack policy can be represented as a finite horizon Markov decision process (MDP). We use domain-specific MDP algorithms to find optimal strategies.[6]

## Game Model

Honeypots in network security are a form of a deception, and we can model deception in games where one player has more information about the game state than the other (such as the network structure or honeypot locations). Predicting player behavior is more difficult in these games because it depends on players' beliefs about the likelihood of possible states. We model the honeypot allocation problem as a chance player who makes an initial random move with a probability corresponding to the attacker's belief of each network's likelihood, as shown in Figure 2. The attacker is uncertain which of the possible extended networks (networks after the chance move) is the defender's real network before the defender added honeypots.

We assume that the attacker's belief is common knowledge (if not, the defender can approximate it by analyzing real-world network frequency). In our example, the core network (at the chance layer) consists of a gateway router and a target host T. We model an attacker who knows that there are two additional hosts, either of host type A or B, with equal probability. Therefore, the chance player extends the core network by adding possible combinations of host types A and B, each with a probability of 1/3 (known to both players), which leads to Networks 1 through 3.

The defender then decides which honeypot host types to add to each extended network. Although the defender knows the actual network, the strategy specifies the honeypots to add to every possible network. We assume that the attacker knows that the defender can add up to $k$ honeypots, therefore, he or she must reason about what the defender would do in each possible situation. This creates an interesting information structure. If the defender adds two honeypots to A in Network 1, which results in Network 1-L, the attacker can be certain that two out of four hosts in A are honeypots. However, if the defender adds one honeypot to A and one to B in Network 1 and two honeypots to A in Network 2, the resulting Networks 1-R and 2-L look exactly the same to the attacker. When the attacker observes this network, he or she can't be certain if the host in B is a honeypot or not. An information set is the set of networks that look the same to the attacker; in Figure 2, they're denoted by dashed rectangles I1 through I5. The attacker must play the same strategy for all networks in an information set because they're indistinguishable. However, in each information set, the attacker can have different attack plans to choose from (for example, in Network 1-L, only host type A can be attacked, while in Network 3-R, only host type B). The defender controls the attacker's observations about the network to a large extent, leading to the potential for deception and a complex decision problem for the attacker.

We assume that the honeypot duplicate is indistinguishable from the original host to the attacker. If the attacker goes after host type A in Network 1-L, he or she has a 50 percent chance of attacking the honeypot. Therefore, the attacker must weigh the probability of being detected against the expected benefit of the successful attack and the cost of alternative attacks. Each attack policy results in a potentially different pair of utility values specified in the leaves of the game tree ($AP_1$ through $AP_n$ in I5 in Figure 2). The first value is the defender's utility, which contains honeypot costs and expected losses from attacks. The second value is the attacker's utility, which contains the expected reward, cost, and penalty for being detected.

Figure 2 shows the defender's strategy $OPT_d$, with probabilities highlighted in blue. The strategy prescribes the probability of playing each action in that network. The defender can influence the attacker's probabilities of observing networks—typically, the best strategies make the attacker indifferent about which host type to attack first, which in turn leads to the least effective attacks. For example, if the defender plays the $OPT_d$ strategy, then the attacker who observes a network in I3 faces with probability $0.45/(0.45 + 0.11) \approx 0.8$ Network 2-R and with probability 0.2 Network 3-L. Because attacks are four times more likely to succeed at host type B than at A, the defender prefers adding a honeypot to B four times more likely than to A.

Computing the defender's strategy is difficult because the number of possible defender actions grows combinatorically with the number of honeypots and host types. Computing the attacker's optimal attack plan is an NP-hard problem, and computing the defender's strategy in Stackelberg equilibrium is an NP-hard problem for imperfect information games. We describe the game model in more detail along with several approximation algorithms elsewhere.[7]

## Case Study I: TV Company

We now present a case study to demonstrate how we can use this framework to model a real network and feasibly compute optimal honeypot allocations. The optimal strategies for this network incorporate deception, with the defender exploiting the attacker's uncertainty about the network.

### Domain Description

The case study is based on a network that the Swedish Defense Research Agency used during its cybersecurity exercises in 2012,[8] in which networks were deliberately left vulnerable to attacks. Here, we simplify the original network by representing complete subnetworks as single host types and reducing the number of vulnerabilities to three PCs and routers. The resulting attack graph has 61 actions and 102 facts.

The attacker can initiate an attack on any PC (via a malicious website visited by a user on that host) except the target studio host (T in Figure 3). The routers can't be attacked remotely, only locally. The attacker aims to gain root privileges to the studio host, which has a value of 1,000 for both players. Exploit actions have costs between 5 and 10 (7.5 on average), and success probabilities are estimated by MulVAL (0.7 on average). The attacker has a penalty of 200 if detected, and the defender's cost for creating a complex honeypot mimicking studio is 130, while any other PC costs 30.

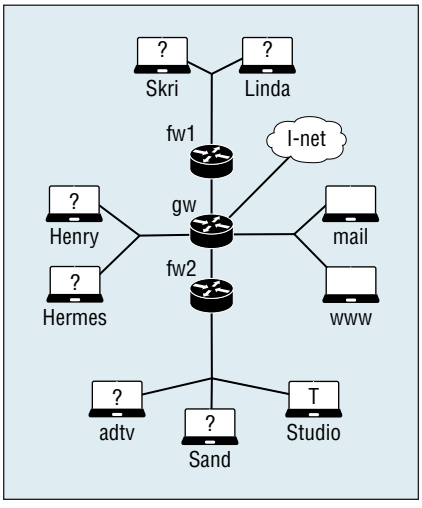


**Figure 3. Network for TV company case study. The attacker knows about all routers, host mail, www, and studio (target) in the network, but he's uncertain if hosts with a question mark are present in the network. Therefore, he must consider all possible combinations and likelihood of their occurrence, to compute optimal attack strategies.**

### Game Analysis

The attacker knows the core network (nodes without question marks in Figure 3). Although uncertain about the hosts with question marks, the attacker knows that the attacked network contains two of the hosts with a question mark, just not which ones exactly. The defender knows his or her network, which consists of the core network and PC host types adtv and Linda. We refer to this as the real network.

Solving the game means finding the defender's optimal strategy for honeypot allocation in every possible network instance that can occur (after the chance move). The defender adds two honeypots of the studio host type to secure the target host, despite their high cost. A more interesting strategy occurs with three honeypots, where the attacker reasons about $\binom{7}{3} = 35$ possible networks and computes attack plans. Because adtv appeals to the attacker, the defender's strategy recommends adding adtv as a honeypot in networks where adtv doesn't exist (that is, in hypothetical networks other than the real one). This makes the attacker cautious about attacking adtv. The defender of the real network (with Linda and adtv) allocates honeypots

- with probability 0.75: $\pi_1$ = {fw2, adtv, studio}, and
- with probability 0.25: $\pi_2$ = {fw2, gw, Skri},

with expected utility –412, which leads to a counterintuitive strategy for the rational attacker, who attacks adtv only when two adtv hosts are present, one real and one honeypot (after action $\pi_1$). Although there's a probability of 0.5 that the attacker will interact with a honeypot, it's worth the risk. Counterintuitively, when the defender plays $\pi_2$, in which case adtv could be attacked without interacting with a honeypot, the attacker reasons that it's "too good to be true" and instead attacks host www. When the attacker observes a single adtv host, the host is more likely to be a honeypot (with a probability of 0.65) than a real host: in alternative networks, adtv is often added as a honeypot.

With three honeypots, the administrator can exploit the attacker's uncertainty by playing mixed strategies, thus reducing total costs. With more honeypots, the optimal strategies become difficult to comprehend in full detail, let alone calculated by hand. Therefore, we argue that a decision support system of this kind is valuable to administrators.

## Case Study II: Human Strategies

We conducted a survey to compare the performance of game-theoretic solutions to human opponents. The 45 respondents who participated in the survey were either the attendees of a four-day forensic malware seminar, members of a multiagent technology group at Czech Technical University in Prague, or employees of computer security companies. The survey was presented as a competition to motivate respondents to create effective strategies.

To avoid overwhelming participants with too much information, we used the simple networks in Figure 2 and set all honeypot costs and attack action costs to zero. We kept the reward of 1,000 for target host T and a penalty of 200 for the attacker if detected, and the exploit success probabilities at 0.2 and 0.8 for host types A and B, respectively. Attacking routers and host type T is always successful, but the attacker can initiate attacks only from host types A or B.

### Respondent Behaviors

Our analysis of 45 collected responses revealed five main clusters in the respondents' defense strategies. We compared this clustering to 500 uniformly random datasets using five standard quality metrics. The quality of our clustering was better than the 95th percentile, indicating that the clusters aren't likely to appear by chance. A strategy belongs to a cluster if its L1 distance from a hand-selected centroid strategy is less than 1/3.

For each cluster, we present the percentage of strategies in that cluster and the defender's average expected utility ($u_d$) of the strategies against a worst-case attacker:

- Optimal strategy ($\text{OPT}_d$), with 15 percent, $u_d = -236 \pm 23$, is a cluster around the game-theoretic solution.
- Perfect information ($\text{PI}_d$), with 26 percent, $u_d = -267 \pm 21$, defends each network in isolation by playing into I1, I4, and I5; it doesn't exploit the attacker's uncertainty.
- Single information set ($\text{SIS}_d$), with 11 percent, $u_d = -343 \pm 16$, always plays to I5.
- Biased uniform ($\text{BU}_d$), with 26 percent, $u_d = -292 \pm 22$, mostly allocates one honeypot to each side, with probabilities 0.1 and 0.2 for both honeypots to A and B, respectively.

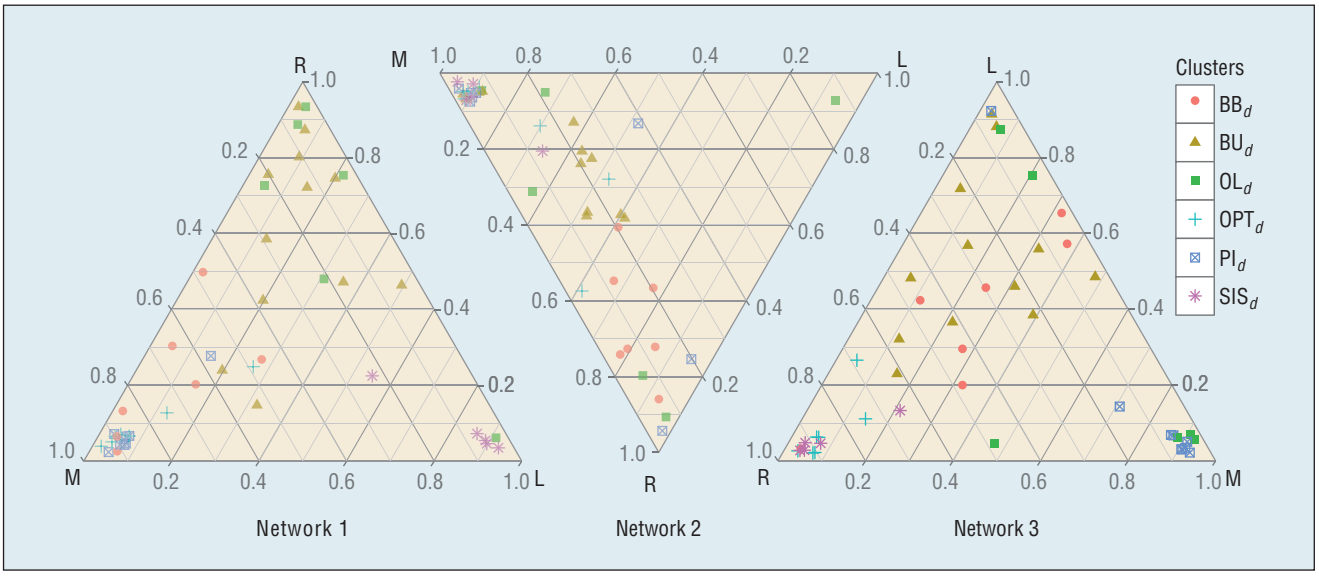


**Figure 4. Respondents' defense strategies. Each triangle represents the space of possible defender strategies for the network in Figure 2 (each corner is a pure strategy).**

**Table 1. Defender's utility, standard deviation, 10th and 90th percentile for the defender's strategies (rows) and attacker's strategies (columns).***

| Defense/attack | Best response (BR$_a$) | | | Mean respondent (MR$_a$) | | |
|---|---|---|---|---|---|---|
| | $\mu$ | $\sigma$ | 10th, 90th percentile | $\mu$ | $\sigma$ | 10th, 90th percentile |
| Optimal (OPT$_d$) | –207 | 131 | –350, –50 | –189 | 181 | –350, 100 |
| Respondent (R$_d$) | –285 | 236 | –500, 100 | –167 | 208 | –360, 100 |
| Mean respondent (MR$_d$) | –262 | 266 | –500, 100 | –164 | 207 | –360, 100 |
| Baseline uniform | –317 | 322 | –800, 100 | –162 | 240 | –500, 100 |
| Best response (BR$_d$) | –302 | 187 | –500, –50 | –111 | 200 | –500, 100 |

* BR$_a$ is the attacker's best-response strategy to a defense strategy; MR$_d$ and MR$_a$ are mean respondent defense and mean respondent attack strategies, respectively.

- Both to B (BB$_d$), with 15 percent, $u_d = -255 \pm 24$, protects the more vulnerable host type B with two honeypots.
- Outliers (OL$_d$), with 13 percent, $u_d = -334 \pm 111$, are defense strategies with a distance larger than 1/3 to any centroid strategy.

Figure 4 illustrates the individual defense strategies. Each triangle represents one possible network, and the points represent the normalized probability distributions over pure strategies (each corner represents a pure strategy).

## Survey Analysis

Table 1 summarizes the strategy analysis. Each entry contains the defender's mean utility $\mu$, standard deviation $\sigma$, and 10th (or 90th) percentile from $10^6$ simulations for a pair of a defense strategy (row) against an attack strategy (column).

The OPT$_d$ strategy maximizes the defender's utility against the worst-case attacker. Column BR$_a$ is the best-response (worst-case) attack strategy against each defense strategy, which shows how exploitable a defense strategy is. R$_d$ represents the respondents' defense strategies in our dataset. We generated a single "mean" respondent strategy by averaging the individual strategies, labeled MR$_d$. By comparing these strategies to the baseline uniform strategy, we see that respondents' play gave better results than the random baseline.

The respondents' mean attack strategy MR$_a$ reveals some interesting observations. The MR$_d$ against MR$_a$ has a lower mean utility than the optimal strategy OPT$_d$ against MR$_a$, indicating that against human attackers, it might be a better defense over OPT$_d$. However, MR$_d$ might not be a good strategy from a long-term perspective. Attackers will likely learn from experience and move toward the best-response BR$_a$ by increasing the frequency of successful attacks and reducing failed ones.[8] Instead of playing OPT$_d$, it might be better to begin with the MR$_d$ strategy, but switch to OPT$_d$ as attackers start adapting.

We can develop an even better defense strategy than OPT$_d$ or MR$_d$ that exploits human behavior. BR$_d$ is the

## THE AUTHORS

**Karel Durkota** is a PhD student and research fellow in the Artificial Intelligence Center at Czech Technical University. His research interests include the defender's decision-making problem using game theory in a complex system. Durkota received an MS in artificial intelligence from Czech Technical University. Contact him at karel.durkota@agents.fel.cvut.cz.

**Viliam Lisý** is a research fellow in the Artificial Intelligence Center at Czech Technical University, where he received his PhD. He's currently a postdoc at the University of Alberta. His research focuses on algorithmic and computational game theory, computing strategies in sequential games, and applications to security. Contact him at viliam.lisy@agents.fel.cvut.cz.

**Christopher Kiekintveld** is an assistant professor at the University of Texas at El Paso. His research interests include intelligent systems, focusing on multiagent systems and computational decision making. Kiekintveld received a PhD in strategic reasoning from the University of Michigan. He has received several best paper awards, the David Rist Prize, and an NSF CAREER award. Contact him at cdkiekintveld@utep.edu.

**Branislav Bošanský** is an assistant professor in the Department of Computer Science at Czech Technical University. His research interests include algorithmic and computational game theory, computing strategies in sequential games, and applications to security. Bošanský received a PhD in artificial intelligence from Czech Technical University. Contact him at branislav.bosansky@agents.fel.cvut.cz.

**Michal Pěchouček** is a full professor at Czech Technical University, where he also heads the Artificial Intelligence Center. His research interests include cyber security, multiagent simulation, and planning. Pěchouček received a PhD in artificial intelligence and biocybernetics from Czech Technical University. Contact him at michal.pechoucek@agents.fel.cvut.cz.

defender's best-response defense strategy against the human $MR_d$ attack strategy, resulting in the highest utility for the defender. However, there's added risk because it's vulnerable against $BR_a$ attackers who specifically exploit this strategy.

Our strategies have a large $\sigma$ due to two factors. First, networks have different levels of security (Network 3 is more vulnerable than Network 1), and more vulnerable networks will naturally have lower utility. For example, with strategy $OPT_d$, the administrator of Networks 1, 2, and 3 has an expected utility –23, –247, and –350, respectively. Second, randomized strategies are necessary to minimize the strategy's predictability, but they also result in variability in outcomes. The $OPT_d$ strategy has the lowest standard deviation against both types of attackers, which can be a desirable property.

Our results show strengths and weaknesses in both theoretical and human solutions: humans were effective at defending against human attackers, but fared poorly against worst-case attackers. Game-theoretic strategies are robust, but there are opportunities to further exploit weaknesses in human opponents. In addition, we had to simplify the game considerably so human players could understand it. In complex scenarios, humans might not be able to compute any plausible strategy with a reasonable effort. Further empirical studies are clearly needed to investigate the effectiveness of game models for network security, but we view this as a promising first step.

Our work has the potential for further research in several directions. The attacker's interaction with honeypots can be modeled in more detail, including the attacker's attempts to detect honeypots. Another direction is to include in the model network changes that can be partially solved by deploying dynamic honeypots. This model could consider sets of possible network changes and find strategies that additionally minimize the cost for honeypot reconfigurations. ■

## Acknowledgments

This work was supported by the Grant Agency of the CTU in Prague (SGS16/235/OHK3/3T/13), Czech Science Foundation (15-23235S), and Cisco Systems.

## References

1. L. Spitzner, "Honeypots: Catching the Insider Threat," *Proc. 19th Annual Computer Security Applications Conf.*, 2003, pp. 170–179.
2. M. Tambe, *Security and Game Theory: Algorithms, Deployed Systems, Lessons Learned*, Cambridge Univ. Press, 2011.
3. X. Ou, W.F. Boyer, and M.A. McQueen, "A Scalable Approach to Attack Graph Generation," *Proc. 13th ACM Conf. Computer and Communications Security*, 2006, pp. 336–345.
4. L. Wang et al., "An Attack Graph-Based Probabilistic Security Metric," *Data and Applications Security*, vol. 5094, 2008, pp. 283–296.
5. P. Mell, K. Scarfone, and S. Romanosky, "Common Vulnerability Scoring System," *IEEE Security & Privacy*, vol. 4, no. 6, 2006, pp. 85–89.
6. K. Durkota et al., "Optimal Network Security Hardening Using Attack Graph Games," *Proc. 24th Int'l Joint Conf. Artificial Intelligence*, 2015, pp. 526–532
7. K. Durkota, "Approximate Solutions for Attack Graph Games with Imperfect Information," *Decision and Game Theory for Security*, Springer, 2015, pp. 228–249.
8. T. Sommestad and F. Sandstrom, "An Empirical Test of the Accuracy of an Attack Graph Analysis Tool," *Information and Computer Security*, vol. 23, no. 5, 2015, pp. 516–531.