Optimal Thresholds for Intrusion Detection Systems

Aron Laszka
University of California, Berkeley
laszka@berkeley.edu

Waseem Abbas
Vanderbilt University
waseem.abbas@vanderbilt.edu

S. Shankar Sastry
University of California, Berkeley
sastry@eecs.berkeley.edu

Yevgeniy Vorobeychik
Vanderbilt University
yevgeniy.vorobeychik@vanderbilt.edu

Xenofon Koutsoukos
Vanderbilt University
xenofon.koutsoukos@vanderbilt.edu

ABSTRACT

In recent years, we have seen a number of successful attacks against high-profile targets, some of which have even caused severe physical damage. These examples have shown us that resourceful and determined attackers can penetrate virtually any system, even those that are secured by the "air-gap." Consequently, in order to minimize the impact of such attacks, defenders have to focus not only on strengthening the first lines of defense but also on deploying effective intrusion-detection systems. Intrusion-detection systems can play a key role in protecting sensitive computer systems, as they give defenders a chance to detect and stop attacks before they could cause substantial losses. However, an over-sensitive intrusion-detection system, which produces a large number of false alarms, imposes prohibitively high operational costs on a defender since every alarm needs to be manually investigated. Thus, defenders have to strike the right balance between maximizing security and minimizing costs. Optimizing the sensitivity of intrusion detection systems is especially challenging in the case when multiple interdependent computer systems have to be defended against a strategic attacker, who can target computer systems in order to maximize losses and minimize the probability of detection. We model this scenario as an attacker-defender security game and study the problem of finding optimal intrusion detection thresholds.

CCS Concepts

•**Security and privacy** → *Intrusion detection systems; Economics of security and privacy;* •**Theory of computation** → *Algorithmic game theory and mechanism design;*

Keywords

Intrusion detection system; game theory; economics of security; Stackelberg equilibrium; computational complexity

HotSoS'16 April 19–21, 2016, Pittsburgh, PA, USA
© 2016 ACM. ISBN .
DOI:

1. INTRODUCTION

After successfully compromising a system, attackers often aim to keep intrusions covert in order to benefit from the defenders' lack of awareness. For example, in cyber-espionage, an attacker's goal is to continue extracting secrets and credentials from its target, which is possible only as long as the intrusion remains undetected. Stealthiness is also crucial to attacking cyber-physical systems in which inalterable characteristics of physical processes can prevent attackers from causing damage immediately after compromising a system. This delay enables defenders to detect and mitigate attacks before the compromised systems suffer significant damage.

However, as attackers strive to be stealthy, security breaches can remain undetected for extended periods of time. For example, the infamous Stuxnet worm reportedly ruined one-fifth of Iran's nuclear centrifuges by subtly increasing the pressure on spinning centrifuges, while showing the control room that everything was normal [13, 16, 14]. As another example, the Maroochy Shire water-services incident lasted several months and was discovered only by accident [2].

To detect stealthy attacks, defenders can deploy *intrusion detection systems* (IDS). An IDS can monitor a computer system or network for signatures of known attacks (e.g., known exploits) or for anomalies (i.e., suspicious activities). For example, an IDS can monitor the system-call traces of critical processes and look for abnormal sequences of system calls, which may be the sign of an intrusion [12, 11]. When an IDS detects suspicious activity, it raises an alarm, which can then be investigated by system operators or security experts.

Unfortunately, practical intrusion detection systems are imperfect. On the one hand, they cannot raise alarms for attacks that do not result in sufficiently suspicious activity and whose signatures are not known. On the other hand, they might raise false alarms for unusual but non-malicious activities. Consequently, the sensitivity of an IDS must be carefully chosen, since too low sensitivity results in excessive losses due to undetected attacks, while too high sensitivity results in wasting resources on investigating false alarms. In an anomaly-based IDS, sensitivity corresponds to a *detection threshold*: activities that are more suspicious than the threshold result in an alarm, while activities that are less suspicious do not.

Finding an optimal detection threshold can prove to be a challenging problem even for a single IDS. However, this problem is much more challenging when IDSes are deployed on multiple computer systems that are independent from a security point of view, but are interdependent with respect to the damage that could be caused by compromising them. For example, in spatially-distributed cyber-physical systems, such electrical grids and water-distribution networks, multiple independent computer systems have control over the same physical process. Since a strategic attacker target will target a subset of these systems by taking into account not only the damage that could be inflicted, but also the probability of remaining undetected, detection thresholds have to be chosen strategically.

In this paper, we study the problem of finding detection thresholds for multiple IDSes in the face of strategic attacks. [1] We model strategic (i.e., rational) attacks against a set of computer systems that are equipped with IDSes as a two-player game between a defender and an attacker. We study the computational complexity of finding optimal attacks and defenses (i.e., optimal detection thresholds) and propose efficient heuristics. Finally, we compare our heuristic IDS thresholds to two baselines using numerical examples based on real-world intrusion detection data. The first baseline, which we call *locally optimal*, is configuring each IDS optimally but independently of the other IDSes. The second baseline, which we call *uniform*, is configuring all the IDSes in the same way, that is, having the same threshold. Our numerical results show that our approach, which optimizes multiple thresholds at the same time, outperforms the baselines, which optimize only one threshold at a time.

The remainder of this paper is organized as follows. In Section 2, we introduce our game-theoretic model and define optimal detection thresholds. In Section 3, we provide theoretical results on our model, and we introduce heuristic algorithms for finding attacks and detection thresholds. In Section 4, we evaluate these algorithms using numerical examples based on real-world intrusion detection data. In Section 5, we discuss related work on intrusion detection thresholds against strategic attacks. Finally, we offer concluding remarks in Section 6.

2. MODEL

In this section, we introduce our game-theoretic model of intrusion detection systems and strategic attacks. For a list of symbols used in this paper, see Table 1.

Table 1: List of Symbols

Symbol	Description
S	set of computer systems to be defended
$FP_s(f_s)$	false-positive rate for system s given that its false-negative probability is f_s
C_s	cost of false alarms for system s
$\mathcal{D}(A)$	damage caused by an undetected attack against the systems in A
$\mathcal{L}(\boldsymbol{f}, A)$	defender's loss for false-negative probabilities $\boldsymbol{f}$ when the attacker targets A
$\mathcal{P}(\boldsymbol{f}, A)$	attacker's payoff for targeting A when the false-negative probabilities are $\boldsymbol{f}$

2.1 Intrusion Detection and Attacker Models

We assume that a defender has to protect a set of computer systems S, each of which is equipped with a host-based *intrusion detection system* (IDS). These IDSes are imperfect in two ways: on the one hand, they might raise an alarm for unusual but normal system behavior, which we call a *false-positive error*; on the other hand, they might fail to raise an alarm when an attack did happen, which we call a *false-negative error.* By changing the *detection threshold* of an IDS, the defender can decrease the rate of false positives and increase the probability of false negatives, or vice versa.

We represent the attainable false-positive rate and false-negative probability pairs for system $s \in S$ as a function $FP_s : [0, 1] \to \mathbb{R}_+$, where $FP_s(f_s)$ is the false-positive rate when the false-negative probability is f_s. We assume that FP_s is a decreasing function, which is indeed true for any practical IDS. Consequently, we will use the terms "detection threshold" and "false-negative probability" interchangeably. Finally, we let vector $\boldsymbol{f}$ denote the false-negative probabilities of all the systems.

When an IDS raises an alarm, the defender has to investigate the system to determine whether an attack has actually taken place. To perform an investigation of system s, the defender has to spend resources (e.g., manpower), which cost her C_s. Consequently, in order to attain false-negative probability f_s for system s, the defender has to waste $C_s \cdot FP_s(f_s)$ on false positives.

We assume that the attacker is capable of mounting an attack against an arbitrary subset of systems (e.g., she has a zero-day vulnerability). The defender will detect and mitigate this attack if the IDS of at least one targeted system raises an alarm. Hence, the probability that an attack targeting a set A of systems will not be detected is

$$\begin{aligned}
&\Pr[\text{attack against set } A \text{ is not detected}] \\
&= \Pr\left[\bigwedge_{s \in A} \text{attack against system } s \text{ is not detected}\right] \quad (1)\\
&= \prod_{s \in A} \Pr[\text{attack against system } s \text{ is not detected}] \quad (2)\\
&= \prod_{s \in A} f_s . \quad (3)
\end{aligned}$$

Finally, an undetected attack will enable the attacker to cause $\mathcal{D}(A)$ damage, where $\mathcal{D} : S \to \mathbb{R}_+$ is a non-decreasing submodular set function.

2.2 Attacker-Defender Game

Next, we formulate the conflict between the defender and the attacker as a leader-follower game and define optimal detection thresholds.

Strategic Choices.

The defender's strategic choice is to select a false-negative probability f_s for each system s by setting the detection threshold (i.e., sensitivity) of its IDS. Recall that the resulting false-positive rate for system s is $FP_s(f_s)$. The attacker's strategic choice is to select a set A of systems to attack.

Defender's Loss and Attacker's Payoff.

When the defender selects false-negative probabilities $\boldsymbol{f}$ and the attacker targets set A, the defender's loss (i.e., in-

[1] Note that IDSes for distributed cyber-physical systems pose other challenging problems as well, e.g., scheduling intrusion detection on resource-bounded devices [1]; however, these problems are beyond the scope of this paper.

verse payoff) is

$$\mathcal{L}(\boldsymbol{f}, A) = \mathcal{D}(A) \prod_{s \in A} f_s + \sum_{s \in S} C_s \cdot FP_s(f_s), \tag{4}$$

that is, the expected amount of damage caused by undetected attacks (i.e., false-negative errors) and the amount of resources wasted on investigating false alarms (i.e., false-positive errors).[2]

For the same strategies $(\boldsymbol{f}, A)$, the attacker's payoff is

$$\mathcal{P}(\boldsymbol{f}, A) = \mathcal{D}(A) \prod_{s \in A} f_s, \tag{5}$$

that is, the attacker benefits from causing damage. The rationale behind this payoff function is the assumption of a worst-case attacker, whose goal is to maximize damage.

Best-Response Attack and Optimal Thresholds.

Following Kerckhoffs's principle, we assume that the attacker knows the defender's algorithms, implementation, etc. and can thus compute the defender's strategy (i.e., the false-negative probabilities chosen by the defender). Hence, the attacker will play a *best response* to the defender's strategy, which is defined as follows.

Definition 1. The attacker's strategy is a *best response* if it maximizes the attacker's payoff, taking the defender's strategy as given. Formally, an attack A is a best response to a given defense strategy $\boldsymbol{f}$ if it maximizes $\mathcal{P}(\boldsymbol{f}, A)$.

On the other hand, the defender cannot respond to the attacker's strategy, and must choose her strategy anticipating that the attacker will play a best response. As is typical in the security literature, we formulate the defender's *optimal* strategy using a refinement of subgame perfect equilibria, called *strong Stackelberg* equilibria [15].

Definition 2. We call a defense strategy *optimal* if it minimizes the defender's loss given that the attacker always plays a best response with tie-breaking in favor of the defender. Formally, an optimal defense is

$$\underset{\substack{\mathbf{0} \leq \boldsymbol{f} \leq \mathbf{1}, \\ A \in \text{bestResponses}(\boldsymbol{f})}}{\operatorname{argmin}} \quad \mathcal{L}(\boldsymbol{f}, A), \tag{6}$$

where bestResponses$(\boldsymbol{f})$ is the set of best-response attacks against $\boldsymbol{f}$.

Note that the effect of the tie-breaking rule is negligible in practice, its sole purpose is to avoid pathological mathematical cases where no optimal strategy would exist.

3. ANALYSIS

Now, we present theoretical results on our model. First, we study best-response attacks in Section 3.1. Then, we investigate optimal detection thresholds in Section 3.2.

3.1 Best-Response Attack

We begin our analysis by studying the computational complexity of finding a best-response attack. To this end, we formulate the problem of finding a best-response attack as a decision problem.

Definition 3. Best-Response Attack Problem (Decision Version) Given a set of computer systems S, false-negative probabilities $\boldsymbol{f}$, a polynomial-time damage function $\mathcal{D}$, and a threshold payoff $\mathcal{P}^*$, determine whether there exists an attack $A \subseteq S$ that attains at least $\mathcal{P}^*$ payoff for the attacker.

The following theorem establishes the computational complexity of finding a best-response attack.

Theorem 1. *Best-Response Attack Problem is NP-hard.*

We prove the above theorem using a reduction from a well-known NP-hard problem, the Maximum Independent Set Problem.

Definition 4. Maximum Independent Set Problem (Decision Version) Given an undirected graph $G = (V, E)$ and a threshold cardinality k, determine whether there exists an independent set of nodes (i.e., a set of nodes such that there is no edge between any two nodes in the set) of cardinality k.

Proof. Given an instance of the Maximum Independent Set Problem (MIS), that is, a graph $G = (V, E)$ and a threshold cardinality k, we construct an instance of the Best-Response Attack Problem (BRA) as follows:

- Let the set of systems be $S := V$.
- Let the false-negative probability for every system $s \in S$ be $f_s := e^{\frac{-1}{k}}$.
- Let the damage function $\mathcal{D}$ be the following. First, establish an arbitrary strict ordering of the set of systems S. Then, for any set A, let $\mathcal{D}$ be the number of systems in A that are independent of the systems in A that precede them.[3]
- Finally, let the threshold payoff be $\mathcal{P}^* := k \cdot e^{-1}$.

Clearly, the above reduction can be performed in polynomial time. Furthermore it is also easy to verify that the function $\mathcal{D}$ defined by the reduction is submodular and polynomial-time computable. Hence, it remains to show that the constructed instance of BRA has a solution *if and only if* the given instance of MIS does.

First, suppose that MIS has a solution, that is, there exists an independent set A of k nodes. We claim that the set A is also a solution to BRA. Since A is independent, the value of $\mathcal{D}(A)$ is equal to the number of systems is A, which is equal to k. Consequently, we have

$$\begin{aligned}
\mathcal{P}(\boldsymbol{f}, A) &= \mathcal{D}(A) \prod_{s \in A} f_s && (7) \\
&= k \prod_{s \in A} e^{\frac{-1}{k}} && (8) \\
&= k \cdot e^{k \frac{-1}{k}} && (9) \\
&= \mathcal{P}^*, && (10)
\end{aligned}$$

which proves that A is a solution to BRA.

Second, suppose that MIS has no solution, that is, every set of at least k nodes is non-independent. Then, we have

[2] Note that we do not explicitly account for resources spent on investigating actual attacks since the cost of these can be incorporated into $\mathcal{D}$.

[3] Note that this function can easily be computed in polynomial time: iterate through the elements of set A according to the ordering, and for each element, test for every preceding element whether an edge exists in the graph.

that $\mathcal{D}(A) < k$ for every A; otherwise, there would exist a set of at least k nodes in A that are independent of each other, which would contradict our supposition. Now, we show that for every $A \subseteq S$, $\mathcal{P}(\boldsymbol{f}, A) < \mathcal{P}^*$. Firstly, for any A such that $|A| \geq k$, we have

$$\begin{align} \mathcal{P}(\boldsymbol{f}, A) &= \mathcal{D}(A) \prod_{s \in A} f_s \tag{11} \\ &< k \prod_{s \in A} e^{\frac{-1}{k}} \tag{12} \\ &\leq k \cdot e^{k \frac{-1}{k}} \tag{13} \\ &= \mathcal{P}^*. \tag{14} \end{align}$$

Note that the inequality is strict. Secondly, for any A such that $|A| < k$, we have

$$\begin{align} \mathcal{P}(\boldsymbol{f}, A) &= \mathcal{D}(A) \prod_{s \in A} f_s \tag{15} \\ &\leq |A| \prod_{s \in A} e^{\frac{-1}{k}} \tag{16} \\ &= |A| e^{\frac{-|A|}{k}}. \tag{17} \end{align}$$

The first derivative of $|A| e^{\frac{-|A|}{k}}$ with respect to $|A|$ is

$$\frac{d}{d|A|} |A| e^{\frac{-|A|}{k}} = e^{\frac{-|A|}{k}} \left(1 - \frac{1}{k}|A|\right). \tag{18}$$

It is easy to see from the above derivative that the maximum of $|A| e^{\frac{-|A|}{k}}$ is attained at $|A| = k$. Consequently, we have that for any A such that $|A| < k$,

$$\mathcal{P}(\boldsymbol{f}, A) = |A| e^{\frac{-|A|}{k}} < k \cdot e^{\frac{-k}{k}} = \mathcal{P}^*. \tag{19}$$

Since the inequality is again strict, we have that BRA cannot have a solution, which concludes our proof. □

As a consequence, unless P = NP, we cannot find a best-response attack in polynomial time. To obtain a near best-response attack, we propose the greedy approach outlined in Algorithm 1. This algorithm starts with an empty set A, and adds systems to the set iteratively. In each iteration, the algorithm chooses an element from $S \setminus A$ that maximally increases the attacker's payoff. If no element increases the attacker's payoff, the algorithm terminates.

Algorithm 1 Greedy Attack

```
Input $S, \boldsymbol{f}, \mathcal{D}$
Initialize: $A \leftarrow \emptyset$, $P^* \leftarrow 0$
while do$A \neq S$
    $s \leftarrow \text{argmax}_{i \in S \setminus A} \mathcal{P}(\boldsymbol{f}, A \cup \{i\})$
    if $\mathcal{P}(\boldsymbol{f}, A \cup \{s\}) > P^*$ then
        $A \leftarrow A \cup \{s\}$
        $P^* = \mathcal{P}(f, A)$
    else
        return $A$
    end if
end while
return $A$
```

Our numerical results show that the greedy algorithm works exceptionally well in practice (see Section 4.2.2). However, in theory, the output of the greedy algorithm could be arbitrarily worse than the best-response attack, as shown by the following proposition.

Proposition 1. *For any $\gamma > 0$, there exists an instance of the Best-Response Attack Problem such that*

$$\frac{\mathcal{P}(\boldsymbol{f}, A^G)}{\mathcal{P}(\boldsymbol{f}, A^*)} < \gamma \tag{20}$$

where A^G is the output of Algorithm 1 and A^* is a best-response attack.

Proof. Consider a set $S = \{1, 2, \ldots, N, N+1\}$ with

$$f_s = \begin{cases} 1 & \text{if } i = \{1, \ldots, N\} \\ 1/N & \text{if } i = N+1, \end{cases} \tag{21}$$

and for any $A \in S$, let $\mathcal{D}(A) = \sum_{i \in A} v_i$, where

$$v_i = \begin{cases} 1 & \text{if } i = \{1, \ldots, N\} \\ N+1 & \text{if } i = N+1. \end{cases} \tag{22}$$

The greedy approach (Algorithm 1) adds system $(N+1)$ to A^G first, since this increases $\mathcal{P}$ to $(N+1)/N > 1$, while adding any other system would increase $\mathcal{P}$ to only 1. Then, the algorithm adds all other systems to A^G as well, since each addition increase $\mathcal{P}$ by $1/N$. Hence, the greedy approach returns the set $A^G = S$, for which the attacker's payoff is $\mathcal{P}(\boldsymbol{f}, A^G) = (2N+1)/N = 2 + 1/N$.

However, the best-response attack is set $A^* = \{1, 2, \ldots, N\}$, for which the attacker's payoff is $\mathcal{P}(\boldsymbol{f}, A^*) = N \cdot 1 = N$. Hence, the ratio between the payoffs is

$$\frac{\mathcal{P}(\boldsymbol{f}, A^G)}{\mathcal{P}(\boldsymbol{f}, A^*)} = \frac{2 + \frac{1}{N}}{N} < \frac{2}{N}. \tag{23}$$

For any $\gamma > 0$, we can let $N = \left\lceil \frac{2}{\gamma} \right\rceil$, so that the ratio is strictly less than γ. □

Next, we introduce another linear-time greedy algorithm for finding an attack, adapted from [5], in Algorithm 2. This algorithm starts with two initial solutions, one containing no element ($X = \emptyset$), and the other containing all elements ($Y = S$). In each iteration, an element $i \in S$ is either added to X, or removed from Y, based on the marginal gains in the attacker's payoff due to adding or removing i. After $|S|$ iterations, the solutions, i.e. X and Y, coincide and a near best-response attack is obtained. Similarly to the previous heuristic, this algorithm also works quite well for practical applications. In fact, it is shown in [5] that this deterministic algorithm gives a 1/3-approximate solution if the objective function is submodular. We note here that the attacker's payoff defined in Equation (5) is not submodular in general, even if $\mathcal{D}(A)$ is submodular function of A.

Algorithm 2 Alternate Linear-Time Attack

```
Input $S, \boldsymbol{f}, \mathcal{D}$
Initialize: $X \leftarrow \emptyset$, $Y \leftarrow S$,
Arrange elements of $S$ in an arbitrary order
for $i = 1$ to $|S|$ do
    $x_i \leftarrow \mathcal{P}(\boldsymbol{f}, X \cup \{i\}) - \mathcal{P}(\boldsymbol{f}, X)$
    $y_i \leftarrow \mathcal{P}(\boldsymbol{f}, Y \cup \{i\}) - \mathcal{P}(\boldsymbol{f}, Y)$
    if $x_i \geq y_i$ then
        $X \leftarrow X \cup \{i\}$
    else
        $Y \leftarrow Y \setminus \{i\}$
    end if
end for
$A \leftarrow X$ (or equivalently $Y$ since $X = Y$)
return $A$
```

3.2 Optimal Detection Thresholds

Now, we study the problem of finding detection thresholds for the defender. First, we provide a necessary condition on the optimal detection thresholds.

PROPOSITION 2. *Let $\boldsymbol{f}$ be optimal false-negative probabilities. Then, for every system s, there exists a set A such that $s \in A$ and A is a best response to $\boldsymbol{f}$.*

PROOF. We prove the claim by contradiction. Suppose that there exists an optimal $\boldsymbol{f}$ such that a system t is not targeted by any best-response attack. Let $\mathcal{P}^*$ be the attacker's payoff for a best response, and let A be a set that contains t and maximizes the attacker's payoff. Then, consider the strategy $\boldsymbol{f}'$ in which f_t is replaced by $\frac{\mathcal{P}^*}{\mathcal{P}(\boldsymbol{f},A)} f_t$. Since FP_t is an increasing function, we have that $\sum_{t \in S} C_t \cdot FP_t(f_t) > \sum_{t \in S} C_t \cdot FP_t(f'_t)$, that is, the total cost of false positives is lower for $\boldsymbol{f}'$ than for $\boldsymbol{f}$. It is also easy to see that every best response to $\boldsymbol{f}$ is also a best response to $\boldsymbol{f}'$. Consequently, the expected amount of losses due to attacks cannot be higher for $\boldsymbol{f}'$ than for $\boldsymbol{f}$, which implies that the defender's loss $\mathcal{L}$ is lower for $\boldsymbol{f}'$ than for $\boldsymbol{f}$. However, this contradicts our initial assumption that $\boldsymbol{f}$ is optimal. Therefore, the original claim must hold. □

Next, we present an algorithm for finding detection thresholds (i.e., false-negative probabilities). In Section 4, we will compare our approach with the two baseline strategies: uniform and locally optimum thresholds. In the *uniform* strategy, all systems are assigned the same false-negative probability, i.e. $f_s = f$, $\forall s \in S$. The value of f is chosen so that the defender's loss (see Equation (4)) is minimized. In the *locally optimum* strategy, for each system s, the false-negative probability f_s is individually optimized. That is, for each system s, the false-negative probability f_s is chosen to minimize $\mathcal{L}(f_s, \{s\}) = \mathcal{D}(\{s\}) f_s + C_s \cdot FP(f_s)$.

As an alternative to these baselines, we propose here an algorithm based on a metaheuristic to find a strategy $\boldsymbol{f}$ that outperforms both the uniform and locally optimum threshold strategies. In particular, we use *simulated annealing* to find a near-optimal solution $\boldsymbol{f}$. The basic idea of this approach is to start with an arbitrary solution $\boldsymbol{f}$, which we then improve iteratively. In each iteration, we generate a new solution $\boldsymbol{f}'$ in the neighborhood of $\boldsymbol{f}$. If the new solution $\boldsymbol{f}'$ is better in terms of minimizing the defender's loss, then the current solution is replaced with the new one. However, in the case $\boldsymbol{f}'$ increases the defender's loss, the new solution replaces the current solution with only a small probability. This probability depends on the difference between the two solutions in terms of loss as well as a parameter commonly referred to as the "temperature," which is a decreasing function of the number of iterations. These random replacements prevent the search from "getting stuck" in a local minimum. The algorithm is presented below as Algorithm 3.

Algorithm 3 Simulated Annealing Algorithm for Defender

```
 1: Input $S$, $\mathcal{D}$, $\boldsymbol{C}$, $k_{\max}$
 2: Initialize: $\boldsymbol{f}$, $k \leftarrow 1$, $T_0$, $\beta$
 3: $A \leftarrow$ Best_Response_Attack($\boldsymbol{f}$)
 4: $L \leftarrow \mathcal{L}(\boldsymbol{f}, A)$
 5: while $k \le k_{\max}$ do
 6:     $\boldsymbol{f}' \leftarrow$ Perturb($\boldsymbol{f}, k$)
 7:     $A' \leftarrow$ Best_Response_Attack($\boldsymbol{f}'$)
 8:     $L' \leftarrow \mathcal{L}(\boldsymbol{f}', A')$
 9:     $c \leftarrow e^{(L'-L)/T}$
10:     if $(L' < L) \lor (\texttt{rand}(0,1) \le c)$ then
11:         $\boldsymbol{f} \leftarrow \boldsymbol{f}'$, $L \leftarrow L'$
12:     end if
13:     $T \leftarrow T_0 \cdot e^{-\beta k}$
14:     $k \leftarrow k + 1$
15: end while
16: return $\boldsymbol{f}$
```

In Algorithm 3, Perturb$(\boldsymbol{f}, k)$ defines the neighborhood of $\boldsymbol{f}$ in the kth iteration, from which $\boldsymbol{f}'$ is randomly chosen. More precisely, in our algorithm, Perturb$(\boldsymbol{f}, k)$ means that each f_s in $\boldsymbol{f}$ is replaced by $f'_s = f_s + \Delta f_s$. Here, for each $s \in S$, Δf_s is randomly picked from the uniform distribution over $\left[-\alpha\left(\frac{k_{\max}-k}{k_{\max}}\right)\ ,\ \alpha\left(\frac{k_{\max}-k}{k_{\max}}\right)\right]$ for some $\alpha \in (0\ \ 1)$. Moreover, since f'_s is a probability, we replace it with 0 if $f'_s < 0$, and replace it with 1 if $f'_s > 1$. Similarly, Best_Response_Attack$(\boldsymbol{f})$ is a routine that computes the attacker's best response for a given $\boldsymbol{f}$, such as Algorithms 1 or 2. In line 13, T is decreasing exponentially with k. We mention here that T could be a linear (or some other) decreasing function of k, but for our application, the exponential function with small values of β (e.g., 10^{-3} for $k_{\max} = 10^4$) works quite well. Finally, we note that a simpler algorithm could also be obtained, in which $\boldsymbol{f}$ is updated with $\boldsymbol{f}'$ in each iteration only if $\boldsymbol{f}'$ is strictly better than $\boldsymbol{f}$. This heuristic search, commonly known as *hill climbing*, also works well for our problem; however, Algorithm 3 gives better results.

4. NUMERICAL ILLUSTRATION

In this section, we evaluate our approach numerically using an IDS based on a real-world dataset and two example instances of our model. Please note that the goal of this effort is not to devise an IDS that performs better than existing ones, since our model assumes that the IDS (with the attainable false-positive rates and false-negative probabilities) is given. In other words, the IDS presented below serves only the purpose of comparing different threshold strategies.

4.1 Intrusion Detection Dataset

We used the ADFA-LD intrusion detection dataset to train and evaluate a host-based IDS [7, 8, 6]. The ADFA-LD dataset is a publicly available collection of system-call traces, which are representative of modern attack structure and methodology, as well as normal system behavior. The dataset consists of three subsets:

- Training data: 833 traces of system calls collected during normal operation, with activities ranging from web browsing to LaTeX document preparation.
- Normal data for validation: 4373 traces of system calls collected the same way as the training data.
- Attack data for validation: 747 traces of system calls collected from various attacks, ranging from exploiting a PHP remote-file inclusion vulnerability to bruteforcing passwords for an SSH service.

Using the ADFA-LD dataset, we trained an IDS based on a simple variant of the approach proposed by Hofmeyr et al. [12]. First, we extracted short, fixed-length sequences of system calls from the training data by sliding a fixed-length window over the system-call traces. Then, we discarded all duplicate sequences, and used the unique sequences from the training data to define the set of normal sequences. Finally, for each system-call trace in the validation sets, we extracted fixed-length sequences of system calls by again sliding a window over the trace, and calculated the ratio of sequences that were abnormal (i..e, sequences that did not appear in the set of normal sequences). If the ratio of abnormal sequences was over a detection threshold, we reported the trace as an attack; otherwise, we reported it as normal behavior. By varying the detection threshold, we could attain various false-positive and false-negative error rates.

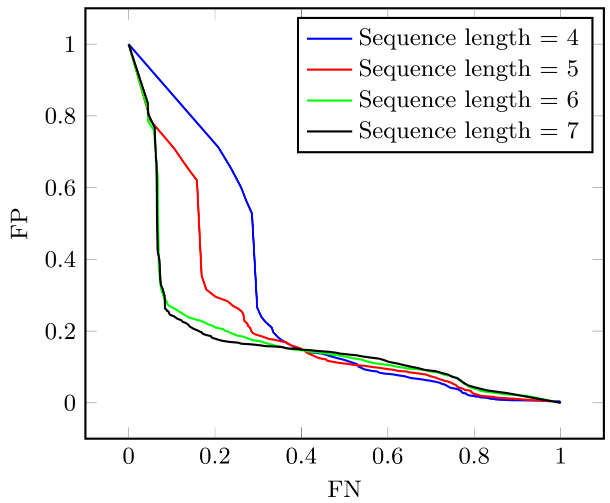

Figure 1: Trade-off between false-negative and false-positive errors in the ADFA-LD dataset for various system-call sequence lengths.

Figure 1 shows the attainable false-positive and false-negative error rates (i.e., fractions of misreported normal and attack traces, respectively) of the IDS for various sequence lengths. The error rates are higher than for IDSes based on more sophisticated algorithms; however, they are comparable [8]. Since our goal is not to devise a novel IDS, but to study the choice of detection thresholds, the curves shown in Figure 1 are suitable for our numerical illustrations. Note that we do not consider sequences longer than 7 because they lead to negligible improvement over shorter sequences.

4.2 Example Systems

Now, we compare the detection thresholds obtained from our approach (Algorithm 3) with the naïve baselines, uniform and locally optimum thresholds, using two examples.

4.2.1 *Basic Example*

First, we study a basic example, which consists of only three computer systems. In this example, we instantiate our model as follows:

- $S = \{a, b, c\}$,
- $\mathcal{D}(A) = 1_{\{a \in A \vee b \in A\}} + 2_{\{a \in A \vee c \in A\}} + 4_{\{b \in A \vee c \in A\}}$,

where $x_{\text{condition}}$ is equal to x if the condition holds and zero otherwise. Since this example consists of only three systems, we can find the best-response attack for any $\boldsymbol{f}$ using an exhaustive search.

Figure 2 shows the defender's loss in Algorithm 3 as a function of the number of iterations. Note that in Figures 2 and 3, we let $C_s = 1$ for every $s \in S$. We can see that there is practically no improvement in loss after 2,000 iterations, which suggests that the solutions are very close to optimal.

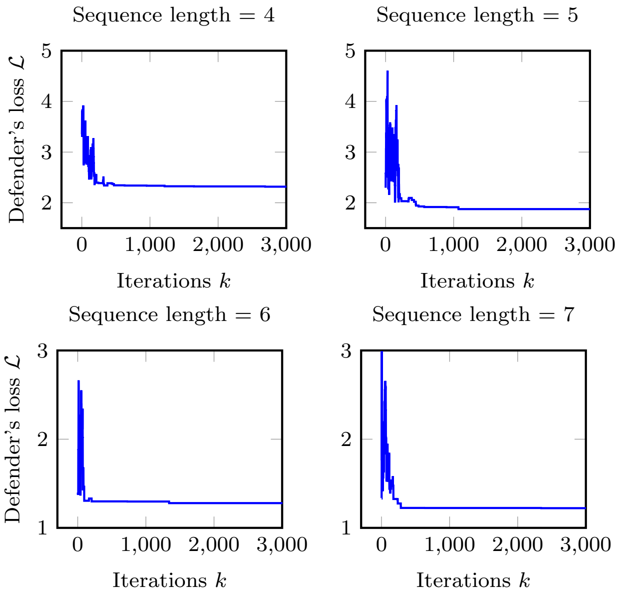

Figure 2: Defender's loss in Algorithm 3 as a function of the number of iterations in the basic example for IDSes based on various sequence lengths.

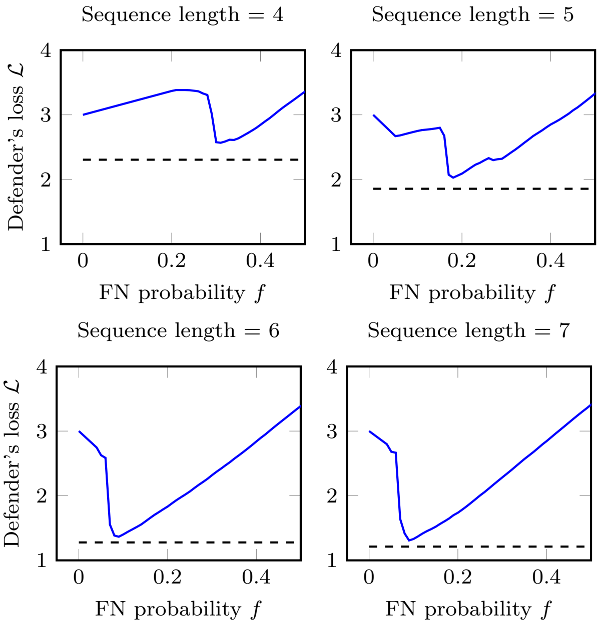

Figure 3: Defender's loss with uniform strategy as a function of false-negative probability in the basic example for IDSes based on various sequence lengths.

Figure 3 shows the defender's loss with the uniform strategy as a function of the false-negative probability f. We can see that even the best uniform strategies perform significantly worse than the strategies found by Algorithm 3, whose loss values are marked by the dashed lines. More specifically, the defender's loss is at least 8% higher for the best uniform strategy than for the output of Algorithm 3.

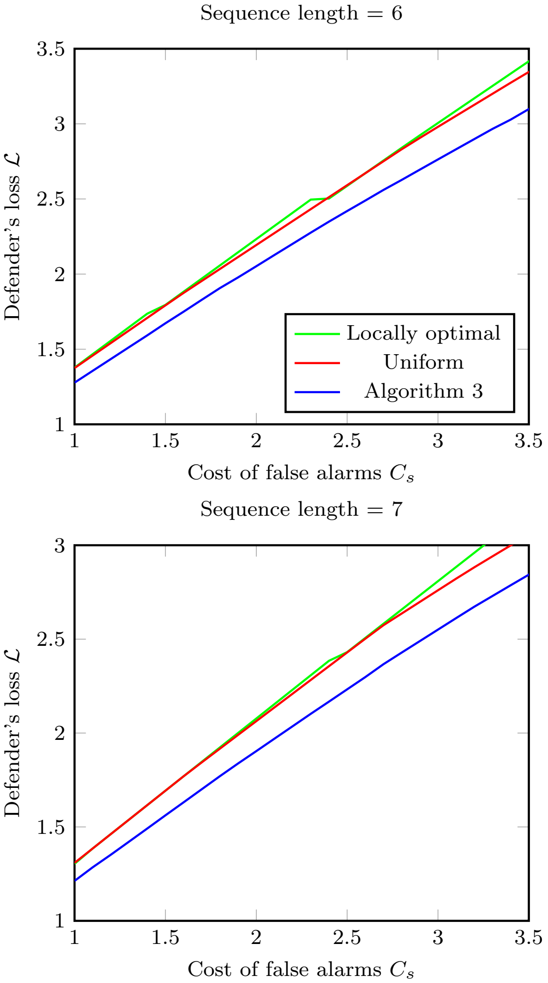

Figure 4: Defender's loss using three different strategies (uniform, locally optimal, and Algorithm 3) as a function of the cost of false alarms in the basic example for IDSes based on various sequence lengths.

Finally, Figure 4 compares the loss values of uniform strategies, locally optimal strategies, and the strategies found by Algorithm 3. We can see that the strategies found by our approach perform substantially better than the baselines.

4.2.2 *Water Distribution Network*

We consider the example of water distribution networks, in which pressure sensors are deployed at various nodes (representing junctions of pipes) to detect changes in pressure owing to pipe leakages and bursts. An attacker may compromise a subset of sensors and alter their true observations. Altering observations enables the attacker to suppress the detection of failures (i.e., pipe bursts), which can result in physical damage and monetary losses, or to fake failures, which can lead to the wastage of resources. To detect intrusions, host-based intrusion detection systems can be installed on the nodes containing pressure sensing devices.

In the distance based threshold model used in the context of sensor placement in water networks [9], the network is represented as a graph, in which nodes correspond to junctions of pipes and links correspond to pipes between junctions. In this model, a sensor deployed at a node can detect the burst of pipes that are at most D distance away from the sensor. The distance between a node and a link is defined as follows: if the link is connected to the node, their distance is 1; otherwise, their distance is 1 plus the length of the shortest path to the node from the end of the link which is closer to the node. In our example, we assume $D = 3$, that is, sensors can detect bursts that are at a distance of at most three from the sensor.

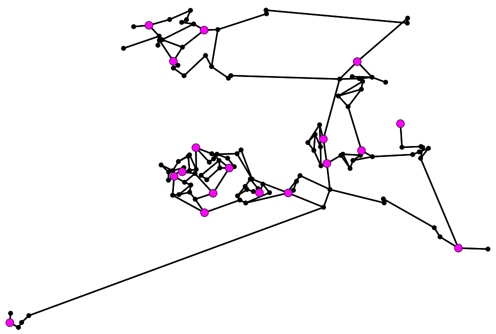

Figure 5: Example water distribution network. Nodes with sensors are highlighted.

In Figure 5, we present a benchmark water distribution network from [19] containing 126 nodes, 168 pipes, one reservoir, one pump, and one storage tank. In our example, sensors at 18 nodes (which are highlighted in the figure) are sufficient to monitor all the pipes. In the case of an attack against a subset A of sensor nodes, correct monitoring of a portion of the network could be compromised. More precisely, pipes that are monitored by the attacked sensors A could not be observed correctly by the sensors in A. We measure the severity of the attack (i.e., $\mathcal{D}(A)$) using the number of pipes whose monitoring is compromised. Formally, we let

- S: set of sensor nodes that need to be defended,
- $\mathcal{D}(A)$: number of pipes (links) that are monitored by the sensors in $A \subseteq S$,
- C_s: cost of investigating a false alarm on sensor s.

We note here that since $\mathcal{D}$ is a coverage function, it is submodular (as we assume in our model in Section 2).

Greedy Attack.

Since the number of systems is higher in this example, we have to use a heuristic algorithm to find an attack instead of an exhaustive search. Here, we demonstrate that the greedy approach presented in Algorithm 1 works well in practice. For this purpose, given some $n \in \{2, 3, \cdots, 10\}$, we select a set S of n sensors in a greedy manner such that they monitor the maximal number of links in the network.

First, we set a uniform false-negative probability for all

Table 2: Comparison Between Best-Response Attacks and the Output of Algorithm 1

n	Fraction of instances where greedy and best-response payoffs are equal	Worst case ratio between greedy and best-response payoffs
2	100%	100%
3	99.9%	97.99%
4	99.5%	93.41%
5	98.2%	86.03%
6	98.1%	85.62%
7	96.1%	75.27%
8	94.9%	82.72%
9	95.2%	82.7%
10	95.7%	77.32%

of the n IDSes. Then, we compute the best-response attack using exhaustive search and compare it with the output of Algorithm 1. We find that for every n, the best-response and greedy attacks' payoffs are exactly the same.

Second, we repeat the same steps, but instead of selecting uniform false-negative probabilities, we pick random $\boldsymbol{f}$. In particular, for each $n \in \{2, 3, \cdots, 10\}$, we generate 1000 instances and compute the best-response and greedy attacks' payoffs for each instance. The results are summarized in Table 2. Again, we observe that for each n, the greedy payoff is equal to the best-response payoff for an overwhelming majority of the instances.

Detection Thresholds.

Next, for the problem of finding detection thresholds, we assume that 18 sensors are deployed in the network to monitor all of the links, and IDSes are deployed on all the nodes with sensors. As before, the objective is to select the thresholds (i.e., false-negative probabilities) of these IDSes to minimize the defender's loss as defined in Equation (4), assuming that the attacker will respond using a greedy attack.

Figure 6 shows the defender's loss in Algorithm 3 as a function of the number of iterations. Note that in Figures 6 and 7, we assume the cost of false alarms to be 1 for every sensor (i.e., $C_s = 1$).

Figure 7 shows the defender's loss with uniform strategies as a function of the false-negative probability f. The dashed line in each plot marks the minimal loss achieved using Algorithm 3. We can see that in every case, the defender's loss with the uniform strategy is at least 18% higher than using the strategy output by Algorithm 3.

Finally, Figure 8 shows a comparison between the naïve strategies and our approach (Algorithm 3) in terms of the defender's loss. We see that our approach clearly outperfs both the uniform and the locally optimal strategies for all values of C_s.

5. RELATED WORK

The problem of setting the sensitivity of an IDS in the presence of strategic attackers has been studied in a variety of different ways in the academic literature. However, to the best of our knowledge, prior work has not considered the problem of simultaneously setting the sensitivity of multiple IDSes that monitor separate but interdependent computer systems. For example, Alpcan and Basar study distributed intrusion detection in access control systems as a security game between an attacker and an IDS, using a model that captures the imperfect flow of information from the attacker to the IDS through a network [3, 4]. The authors investi-

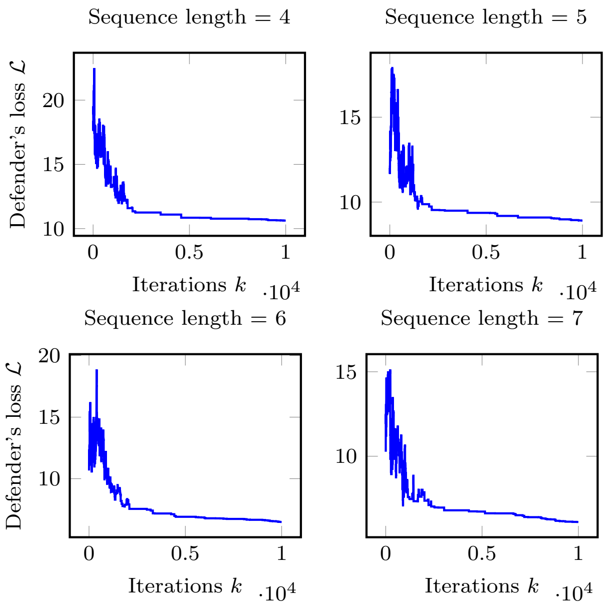

Figure 6: Defender's loss in Algorithm 3 as a function of the number of iterations in the water-distribution network for IDSes based on various sequence lengths.

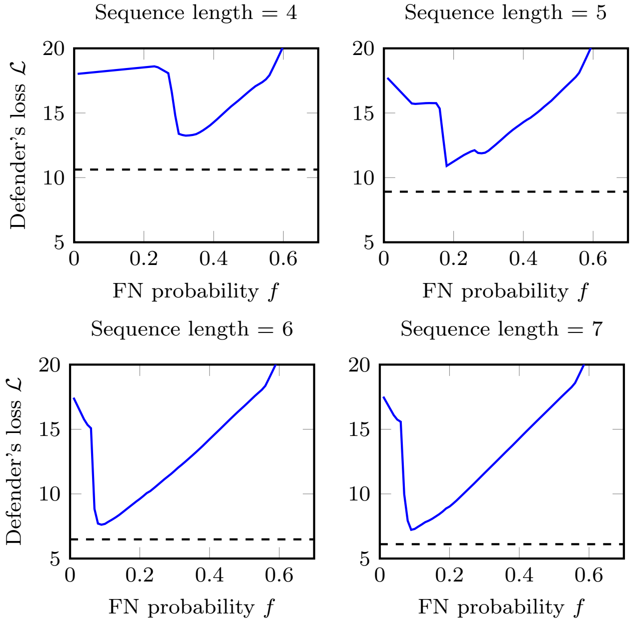

Figure 7: Defender's loss with uniform strategy as a function of false-negative probability in the water-distribution network for IDSes based on various sequence lengths.

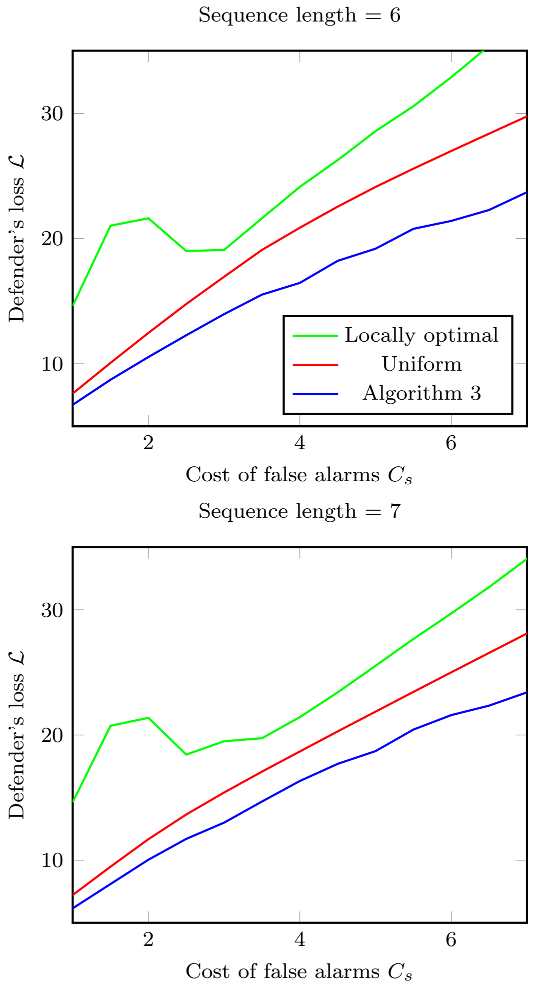

Figure 8: Defender's loss using three different strategies (uniform, locally optimal, and Algorithm 3) as a function of the cost of false alarms in the water-distribution network for IDSes based on various sequence lengths.

gate the existence of a unique Nash equilibrium and best-response strategies under specific cost functions, and analyze long-term interactions using repeated games and a dynamic model. As another example, Dritsoula et al. consider the problem of setting a threshold for classifying an attacker into one of two categories, spammer and spy, based on its intrusion attempts [10]. They give a characterization of the Nash equilibria in mixed strategies, and show that the equilibria can be computed in polynomial time. More recently, Lisỳ et al. study randomized detection thresholds using a general model of adversarial classification, which can be applied to e-mail filtering, intrusion detection, steganalysis, etc. [18]. The authors analyze both Nash and Stackelberg equilibria based on the true-positive to false-positive curve of the classifier, and find that randomizing the detection threshold may force a strategic attacker to design less efficient attacks.

The strategic selection of thresholds for filtering spear-phishing and other malicious e-mail is also closely related to the problem considered in this paper. Laszka et al. study a single defender who has to protect multiple users against targeted and non-targeted malicious e-mail [17]. The authors focus on characterizing and computing optimal filtering thresholds, and they use numerical results to demonstrate that optimal thresholds can lead to substantially lower losses than naïve ones. Zhao et al. study a variant of the previous model: they assume that the attacker can mount an arbitrary number of costly spear-phishing attacks in order to learn a secret, which is known only by a subset of the users [20, 21]. They also focus on the computational aspects of finding optimal filtering thresholds; however, their variant of the model does not capture non-targeted malicious e-mails, such as spam.

6. CONCLUSION

Intrusion detection systems play a key role in securing computer systems against stealthy attacks. In this context, optimizing the sensitivity of IDSes by tuning their detection thresholds is crucial to maximizing security while minimizing costs, which might be incurred as a consequence of raising false alarms or ignoring actual attacks. In this direction, we modeled strategic attacks and optimal intrusion detection strategies using the game-theory nomenclature. We then proposed heuristic algorithms to find strategic attacks and to select detection thresholds for IDSes. Using a basic example as well as a case study of a water distribution network, we compared our algorithm for selecting detection thresholds with the two baseline strategies, optimal uniform strategy and locally optimal strategy. The numerical results showed that our approach outperforms the baseline strategies in terms of minimizing the defender's overall losses.

In future work, we aim to extend this work by considering other classes of damage functions, such as supermodular and additive functions, to accommodate a wider variety of applications. Another direction we wish to pursue is to exploit the application-specific characteristics of the systems to optimize the deployment of IDSes and tune their detection thresholds to maximize security while minimizing losses.

7. REFERENCES

[1] W. Abbas, A. Laszka, Y. Vorobeychik, and X. Koutsoukos. Scheduling intrusion detection systems in resource-bounded cyber-physical systems. In *Proceedings of the 1st ACM Workshop on Cyber-Physical Systems Security and Privacy (CPS-SPC)*, pages 55–66, October 2015.

[2] M. Abrams and J. Weiss. Malicious control system cyber security attack case study – Maroochy Water Services, Australia. http://csrc.nist.gov/groups/SMA/fisma/ics/documents/Maroochy-Water-Services-Case-Study_report.pdf, Jul 2008.

[3] T. Alpcan and T. Basar. A game theoretic approach to decision and analysis in network intrusion detection. In *Proceedings of the 42nd IEEE Conference on Decision and Control (CDC)*, volume 3, pages 2595–2600. IEEE, 2003.

[4] T. Alpcan and T. Başar. A game theoretic analysis of intrusion detection in access control systems. In *Proceedings of the 43rd IEEE Conference on Decision and Control (CDC)*, volume 2, pages 1568–1573. IEEE, 2004.

[5] N. Buchbinder, M. Feldman, J. S. Naor, and R. Schwartz. A tight linear time (1/2)-approximation for unconstrained submodular maximization. *SIAM Journal on Computing*, 44(5):1384–1402, 2015.

[6] G. Creech. *Developing a high-accuracy cross platform Host-Based Intrusion Detection System capable of reliably detecting zero-day attacks*. PhD thesis, University of New South Wales, 2014.

[7] G. Creech and J. Hu. Generation of a new IDS test dataset: Time to retire the KDD collection. In *Proceedings of the 2013 IEEE Wireless Communications and Networking Conference (WCNC)*, pages 4487–4492, 2013.

[8] G. Creech and J. Hu. A semantic approach to host-based intrusion detection systems using contiguousand discontiguous system call patterns. *IEEE Transactions on Computers*, 63(4):807–819, 2014.

[9] A. Deshpande, S. E. Sarma, K. Youcef-Toumi, and S. Mekid. Optimal coverage of an infrastructure network using sensors with distance-decaying sensing quality. *Automatica*, 49(11):3351–3358, 2013.

[10] L. Dritsoula, P. Loiseau, and J. Musacchio. Computing the Nash equilibria of intruder classification games. In *Proceedings of the 3rd International Conference on Decision and Game Theory for Security (GameSec)*, pages 78–97. Springer, Nov 2012.

[11] S. Forrest, S. Hofmeyr, and A. Somayaji. The evolution of system-call monitoring. In *Proceedings of the 24th Annual Computer Security Applications Conference (ACSAC)*, pages 418–430, 2008.

[12] S. A. Hofmeyr, S. Forrest, and A. Somayaji. Intrusion detection using sequences of system calls. *Journal of Computer Security*, 6(3):151–180, 1998.

[13] Kaspersky Lab. Kaspersky Lab provides its insights on Stuxnet worm. http://www.kaspersky.com/about/news/virus/2010/Kaspersky_Lab_provides_its_insights_on_Stuxnet_worm, Sep 2010. Accessed: January 20th, 2016.

[14] M. B. Kelley. The Stuxnet attack on Iran's nuclear plant was 'far more dangerous' than previously thought. *Business Insider*, http://www.businessinsider.com/stuxnet-was-far-more-dangerous-than-previous-thought-2013-11, Nov 2013. Accessed: June 21st, 2015.

[15] D. Korzhyk, Z. Yin, C. Kiekintveld, V. Conitzer, and M. Tambe. Stackelberg vs. Nash in security games: An extended investigation of interchangeability, equivalence, and uniqueness. *Journal of Artificial Intelligence Research*, 41(2):297–327, 2011.

[16] D. Kushner. The real story of Stuxnet. *IEEE Spectrum*, 50(3):48–53, 2013.

[17] A. Laszka, Y. Vorobeychik, and X. Koutsoukos. Optimal personalized filtering against spear-phishing attacks. In *Proceedings of the 29th AAAI Conference on Artificial Intelligence (AAAI)*, pages 958–964, Jan 2015.

[18] V. Lisỳ, R. Kessl, and T. Pevnỳ. Randomized operating point selection in adversarial classification. In *Proceedings of the 2014 European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD), Part II*, pages 240–255. Springer, Sep 2014.

[19] A. Ostfeld, J. G. Uber, E. Salomons, J. W. Berry, W. E. Hart, C. A. Phillips, J.-P. Watson, G. Dorini, P. Jonkergouw, Z. Kapelan, et al. The battle of the water sensor networks (bwsn): A design challenge for engineers and algorithms. *Journal of Water Resources Planning and Management*, 134(6):556–568, 2008.

[20] M. Zhao, B. An, and C. Kiekintveld. An initial study on personalized filtering thresholds in defending sequential spear phishing attacks. In *Proceedings of the 2015 IJCAI Workshop on Behavioral, Economic and Computational Intelligence for Security*, Jul 2015.

[21] M. Zhao, B. An, and C. Kiekintveld. Optimizing personalized email filtering thresholds to mitigate sequential spear phishing attacks. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI)*, Feb 2016.