Near-Optimal Node Blacklisting in Adversarial Networks

Christos Dimitrakakis[1] and Aikaterini Mitrokotsa[1]

EPFL, Lausanne, Switzerland
first-name.last-name@epfl.ch

Abstract. Many communication networks contain nodes which may misbehave, thus incurring a cost to the network operator. We consider the problem of how to manage the nodes when the operator receives a payoff for every moment a node stays within the network, but where each malicious node incurs a hidden cost. The operator only has some statistical information about each node's type, and never observes the cost. We consider the case when there are two possible actions: removing a node from a network permanently, or keeping it for at least one more time-step in order to obtain more information. Consequently, the problem can be seen as a special type of intrusion response problem, where the only available response is blacklisting. We first examine a simple algorithm (HIPER) which has provably good performance compared to an oracle that knows the type (honest or malicious) of each node. We then derive three other approximate algorithms by modelling the problem as a Markov decision process. To the best of our knowledge, these algorithms have not been employed before in network management and intrusion response problems. Through experiments on various network conditions, we conclude that HIPER performs almost as well as the best of these approaches, while requiring significantly less computation.

1 Introduction

We consider a communication network which is being partially monitored by a network management system. The nodes in this network can be of one of two types: malicious (e.g. dropping packets), or honest. We assume that we have some tangible gain for every moment that an honest node remains in the network, while keeping malicious nodes in the network carries a cost, both of which are hidden from the system. The system maintains some statistics about the activity of each node, which enable it to approximately guess its type. These could be alerts gathered from some intrusion detection system (IDS). Another example would be data-sharing statistics from a peer-to-peer (P2P) network, which would help identify selfish nodes. We require a response mechanism that removes malicious nodes as soon as possible, without inadvertently removing honest nodes, with high probability. In this setting, immediately removing nodes that seem suspicious, is suboptimal: if a node is removed from the network then no further information can be received from this node. Thus, it pays to delay removal until we collect more information about suspicious nodes.

Our first contribution is a decision-theoretic approach based on distribution-free high probability bounds. The bounds require very little prior information and can be used to trade off the cost of removing honest nodes with that of keeping malicious nodes in the network for too long. We prove that this High Probability Efficient Response algorithm (HiPER) has low *worst-case expected loss* relative to an oracle which knows the type of every node.

Our second contribution is a set of Bayesian decision-theoretic approaches that we derive by formalising the problem as a Markov decision process. These require some further assumptions. In particular, it is necessary to fully specify a structure and prior parameters for the underlying statistical model. In addition, making optimal decisions according to such models is in our case computationally intractable. Consequently, we consider some approximate algorithms. Of these, an optimistic approximation has similar performance to that of HiPER. Another approximation, which performs finite lookahead, can obtain some further performance gain, at the cost of additional computation.

The paper is organised as follows. In the remainder of this section we give some background, present the related work and describe our contributions. Section 2 introduces notation while Section 3 specifies the loss model. Section 4 presents the proposed HiPER algorithm as well as the bounds on the *worst-case expected loss*. Section 5 describes the decision-theoretic approaches which model the problem as an MDP and are used in the performance comparisons with the HiPER algorithm while Section 6 describes the evaluation experiments. Finally, Section 7 concludes the paper. The appendix contains proofs of some technical lemmas and provides some useful auxiliary results for completeness.

1.1 Background

The problem we consider falls within the scope of (statistical) decision theory. In particular, the particular scenario we investigate can be reduced to the *optimal stopping* problem [8], which can be modelled as an (unknown) Markov Decision Process [8] (MDP) or as a (potentially unknown) Partially Observable MDP (POMDP) [18].

More precisely, in our setting, the nodes can be one of two types: honest or malicious. However, we initially start out without knowing what type each node is. Consequently, we must gather data (observations) to reduce our uncertainty about their types. Unfortunately, we can only do so while a node remains within the network. However, the longer we maintain a malicious node in the network, the more loss we incur. Conversely, once we remove an honest node, we will obtain no more profit from it. Thus, the problem can be reduced to deciding at what time, or under which conditions, to remove a given node from the network, if at all. Consequently, our scenario can be seen as a type of *optimal stopping* problem.

Optimal stopping problems can be seen as MDPs [8]. An MDP models the interaction between an environment and an agent. The environment has a state, which changes over time and whose next value depends on the current state, as well as the current action taken by the agent. In addition, at every time-step, the

agent receives a reward, which also depends on both the state and the agent's action. His goal is to maximise his total expected reward. Intrusion response can be modelled as an *unknown MDP*. where each node in the network has an unknown type (malicious or honest), and where the state describes the set of nodes which remain in the network. In our case, the responses that we make correspond to the actions of the agent. However, the *reward* we receive is *hidden* and depends on the unknown type of the nodes. Equivalently, we can view the nodes' type as a hidden state, in which case the intrusion response problem can be seen as a partially observable MDP (POMDP).[1]

To cast our problem in this setting, we need the following elements: a) the *prior* probability for each node being *honest* or *malicious*; b) a *known* distribution family for the observation distribution, conditioned on whether the node under consideration is honest or malicious; c) a planning algorithm that will determine our responses. In general, this can be quite demanding computationally, as the solution to either the unknown MDP or the POMDP problem requires performing planning in a large tree. Finally, since in our case the reward remains hidden from the agent, the problem is an extreme case of a partial monitoring problem [7].

1.2 Previous work

The stopping problem has been extensively studied in general [8], while partial monitoring games in general have also received a lot of attention recently [7]. However, to the best of our knowledge, the general hidden reward stopping problem has not been previously studied in the literature. On the other hand, the specific application we consider can be seen as a type of *optimal intrusion response*, for which there is a considerable body of work.

Most of the previous research on intrusion response has concentrated on the POMDP formalism. Indicative publications are those by Zonouz *et al.* [22], Zan *et al.*, [20] and Zhang *et al.* [21], which have all proposed an intrusion response through modelling the process as a POMDP [18]. More precisely, Zonouz *et al.* [22] proposed a Response and Recovery Engine (RRE) based on a game-theoretic response strategy against adversaries modelled as opponents in a non-zero-sum, two-player Stackelberg stochastic game. In each step of the game RRE chooses the response actions using an approximate POMDP solver. More precisely, using the most likely state (MLS) [6] approximation, the POMDP is converted to a competitive Markov Decision Process (MDP), which is then solved using a look ahead search (i.e. approximate planning). Zhang *et al.* use the POMDP to integrate low level IDS alerts with high level system states, while Zan *et. al.* [20] propose to solve the intrusion response problem as a factored POMDP model. Additionally, they decompose the POMDP into small sub-POMDPs and compute the response policy using the MLS approximation technique. However,

[1] This is not contradictory, since unknown MDP problems are in fact a special case of POMDPs[11] where the unknown parameters are seen as an unchanging, hidden state.

in our case MLS as an approximation is too crude to be used, since it would essentially result in a completely random policy, as there are only two possible hidden states each node can be in. An entirely different approach, policy-gradient methods, is employed by [9] in the context of combating denial-of-service attacks in P2P networks. However, this approach requires observing the rewards, which are in fact hidden in our case.

1.3 Our contributions

Our first proposed algorithm relies on bounds which do not require knowledge of prior probabilities regarding the type of a node (honest or malicious) neither known distributions for the observations corresponding to honest or malicious nodes. We only need to know the mean of each of these distributions. Consequently, it is substantially more lightweight than MDP solvers, since we take decisions without performing explicit planning. Thus, it is more suitable for resource constrained environments such as wireless communication networks. We analyse the *expected loss* of this algorithm, and show that it is not significantly worse to that of an oracle which already knows each node's type.

Our second contribution is to derive three approximate MDP solvers for this problem. In contrast to previous work, in our scenario the reward is *never observed* by the algorithm.[2] This is necessary, since knowing that a node gave you positive reward allows you to directly conclude that it is an honest node. Furthermore, two of our MDP algorithms are fundamentally different from those previously employed in the intrusion response literature, as we forego the most-likely-state approximation commonly used in POMDP-based approaches. The first approach we consider is a myopic approximation. This is equivalent to the most likely state approximation and ot a sequential probability ratio test under some conditions. The second approach is a lightweight *optimistic* heuristic that performs no planning, which is derived from upper bounds [10] on Bayesian decision making in unknown MDPs [11]. To our knowledge, this approach has not been used in similar problems before. Finally, we consider online planning with finite lookahead [15,8]. This approach takes decisions which consider the impact of all our possible future actions up to some horizon. While this approach has not been considered for intrusion response problems before, we note that it has been employed in other applications such as dialogue modelling [5], autonomous underwater vehicle mapping [16], preference elicitation [3] and sensor scheduling [12] in wireless sensor networks.

2 Preliminaries

We consider a network composed of a set of nodes, which can be either honest or malicious. In this work, we ignore the specific network topology, beyond assuming that there is a reliable way to obtain some statistics from each node. We denote

[2] Although of course the reward is used in the experiments to measure performance.

by $\mathcal{Q}$ the set of all malicious nodes and by $\mathcal{U}$ the set of all honest nodes. We consider that there is an entity $\mathcal{E}$ (for instance an Internet Service Provider (ISP) or a network administrator) who gains some reward (gain) $g_{\mathcal{U}}$ from each moment that a honest node remains in the network and has a cost (loss) $\ell_{\mathcal{Q}}$ for each moment thata malicious node stays in the network. A node may be removed by $\mathcal{E}$ at any time, for example through black-listing. However, re-inserting a removed node is not normally possible.

We use N to denote the (possibly random) time at which $\mathcal{E}$ removes a node from the network. In addition, any honest node may leave the network at some (random) time H. Specifically, we assume that an honest node may decide to leave the network with some small probability $\lambda > 0$, independently over time. Then it holds that $\mathbb{E}[H] = \frac{1}{\lambda}$.

Assumption 1 *We assume that each node has a fixed type (i.e. honest or malicious) that is not changing over time. The type is hidden from $\mathcal{E}$.*

$\mathcal{E}$ not only does not know the type of each node, but it also never observes the rewards obtained or the cost incurred. However, at each time-step t and for each node i, $\mathcal{E}$ receives some information signal $x_{i,t} \in [0, 1]$, characterising the behaviour of that node i within the time interval $t \in \mathbb{N}$.

Assumption 2 *We assume that $x_{i,1}, \ldots, x_{i,t}$ are independent,*[3] *but not necessarily identically distributed, random variables and it holds:*

$$\mathbb{E}[x_{i,t} \mid \mathcal{Q}] = q, \quad \mathbb{E}[x_{i,t} \mid \mathcal{U}] = u. \tag{1}$$

It is important to while the expected value is constant for all t, the observed average of $\frac{1}{t}\sum_{k=1}^{t} x_{i,t}$ for each node i will initially be far from the expected value for small t. The average, together with the total number of observations for each node form a summary of the information received by each node. The relationship between these quantities will be looked at more closely in the analysis.

Finally, we place no specific meaning to q and u in this work, as they are application-dependent. In an *intrusion response* scenario (e.g. [13]), they could be considered as the *detection rate* (DR) and the *false alarm rate* (FA) correspondingly of an employed intrusion detection system. Then $x_{i,t}$ would correspond to alarm signals, with lower and high values for innocent-looking and suspicious behaviour respectively. Correspodingly, in a *p*eer-to-peer scenario (e.g. [17]), they could be fairness or reputation scores of each node.

In the remainder, we always refer to some arbitrary node in the network and thus make no distinction between nodes. This is because the algorithms that we examine, consider each node *independently* of the others. Consequently, the following section analyses the expected loss for a single node of unknown type.

[3] This assumption could in perhaps be relaxed if it is possible to form a martingale difference sequence from the sequence of observations.

3 The loss model

As previously mentioned, $\mathcal{E}$ obtains a small gain for each time-step an honest node is within the network, and a small loss for each time-step a malicious node remains in the network. Formally, we can write that the total gain G we obtain from some node i, which $\mathcal{E}$ removes at time N, and which would voluntarily leave at time H is:

$$G(i, H, N) = \begin{cases} -N\ell_{\mathcal{Q}}, & i \in \mathcal{Q} \\ \min\{H, N\} g_{\mathcal{U}}, & i \in \mathcal{U}. \end{cases} \tag{2}$$

$\mathcal{E}$ wants to choose some node removal policy π that maximises his total expected gain. That means that $\mathcal{E}$ needs to keep as many as possible honest nodes in the network and eliminate the nodes that behave maliciously. In our analysis, we compare the expected gain of our policy π with that of an *oracle*. The oracle always knows the type of each node (i.e. honest or malicious), and thus, employs the optimal policy π^*. For $i \in \mathcal{Q}$, according to the optimal policy π^* it holds $N = 0$, while for $i \in \mathcal{U}$ according to the optimal policy π^* it is $N = \infty$. Correspondingly,

$$\mathbb{E}_{\pi^*}[G(i)] = \begin{cases} 0, & i \in \mathcal{Q} \\ \mathbb{E}[H] g_{\mathcal{U}}, & i \in \mathcal{U}. \end{cases} \tag{3}$$

Let the loss L be the difference between the gain of the optimal policy and our policy. In particular, the *expected loss* of policy π for a node of type v is defined as:

$$\mathbb{E}_{\pi}[L \mid v] = \mathbb{E}_{\pi^*(v)}[G \mid v] - \mathbb{E}_{\pi}[G \mid v], \tag{4}$$

where the i subscript has been dropped for simplicity. The expected loss is bounded by the *worst-case expected loss*:

$$\mathbb{E}_{\pi}[L] \leq \max_{v \in \{\mathcal{Q}, \mathcal{U}\}} \mathbb{E}_{\pi}[L \mid v], \tag{5}$$

which we wish to minimise. If $\mathcal{E}$ removes node i from the network at random time N, then he does not receive any more observations $x_{i,t}$ for this node from the IDS. Thus, in essence, we want to find a *stopping rule*, that will let $\mathcal{E}$ to determine the random time N at which stopping occurs, i.e. $\mathcal{E}$ takes the decision that $i \in \mathcal{Q}$ and removes it from the network. We note that, $0 \leq N \leq \infty$, where $N = \infty$ if stopping never occurs.

Since $\mathcal{E}$ does not know if node i is honest or malicious, it must collect a sufficient number of samples so as to only remove nodes for which it is reasonably certain that they are malicious. On the other hand, malicious nodes must be removed as soon as possible, since the operator incurs a cost for every moment they remain in the network. The first algorithm we consider uses simple statistics to make nearly optimal decisions about which nodes to keep.

4 The HiPER Algorithm

The algorithm, depicted in Alg. 1, uses the knowledge we have about malicious and honest nodes (see equation 1). This is done by calculating the average of all the observations generated by a node i until time t:

$$\theta_t \triangleq \frac{1}{t}\sum_{k=1}^{t} x_{i,k}, \tag{6}$$

and adding an appropriate *confidence interval* so that errors are made with low probability. Informally, HiPER keeps nodes in the network as long as the statistic θ_t is sufficiently far from the expected statistic q of malicious nodes. In order to avoid throwing away honest nodes prematurely, it always keeps nodes for a certain number of steps to obtain more reliable statistics. However, as time passes, it needs more and more evidence to kick a node out. Consequently, the probability that an honest node is thrown out is bounded.

The analysis of the algorithm proceeds in three steps. First, we calculate the expected loss of the algorithm when faced with a node of malicious type. Then, we calculate the loss for honest nodes. Subsequently, we combine the two losses and tune the algorithm's input parameters to obtain an overall loss bound.

Algorithm 1 HiPER Algorithm for Optimal Response

Parameters: $\delta, \Delta, q \in [0,1]$
Loop: For each node i in the network:
  For each time-step t do:
    if $|\theta_t - q| < \sqrt{\frac{\ln(2/\delta)}{2t}}$ and $t > \frac{\ln(2/\delta)}{2\Delta^2}$ **then**
      remove node i from the network
    else keep node i in the network.
    end if
  end For
end For

The first bound only depends upon the input parameter δ, the error probability we wish to accept, and the loss $\ell_{\mathcal{Q}}$ incurred by malicious nodes. We prove that the expected loss is polynomially bounded in terms of both δ and $\ell_{\mathcal{Q}}$.

Lemma 1. *For Algorithm 1, with input parameter δ, and $\Delta = |u - q|$, the* expected loss *when the node is malicious is bounded as:*

$$\mathbb{E}[L \mid \mathcal{Q}] \leq \frac{\ell_{\mathcal{Q}}}{(1-\delta)^2} \tag{7}$$

The proof of this lemma can be found in the appendix. Naturally, the expected loss is linearly dependent on the loss of keeping a malicious node in the network, while the dependence on the error probability is quadratic.

The second bound depends on the input parameter Δ, which corresponds to how far we expect the statistics of honest nodes to be from q, the gain obtained by honest nodes $g_{\mathcal{U}}$ and the leaving probability of honest nodes λ. Once more, we obtain a polynomial loss bound in terms of those variables.

Lemma 2. *If $\Delta = |u - q|$, then the* expected loss *when the node is honest is bounded by:*

$$\mathbb{E}[L \mid \mathcal{U}] \leq \frac{g_{\mathcal{U}}(\Delta^2 + 2)}{\lambda(\Delta^2 + 2\lambda)}. \tag{8}$$

The proof of this lemma can be found in the appendix. Similarly to the previous lemma, there is a linear dependence on the loss that is incurred when we erroneously remove an honest node, and a quadratic dependence on the rate of departure. In addition, there is a weak dependence on the gap Δ between the two means.

Finally, we can combine everything in one bound by selecting a value for δ that depends on Δ and which simultaneously makes the bounds tight:

Theorem 1. *Set $\Delta = |u - q|$ and select:*

$$\delta = 1 - \sqrt{\frac{\ell_{\mathcal{Q}} \lambda(\Delta^2 + 2\lambda)}{g_{\mathcal{U}}(\Delta^2 + 2)}} \tag{9}$$

then the expected loss $\mathbb{E}\, L$ is bounded by:

$$\mathbb{E}(L) \leq \mathcal{L}_1 \triangleq \frac{g_{\mathcal{U}}(\Delta^2 + 2)}{\lambda(\Delta^2 + 2\lambda)}. \tag{10}$$

Proof. If we substitute (9) in (4) we get:

$$\mathbb{E}[L|\mathcal{Q}] \leq \frac{\ell_{\mathcal{Q}}}{(1-\delta)^2} = \frac{\ell_{\mathcal{Q}}}{\frac{\ell_{\mathcal{Q}} \lambda(\Delta^2+2\lambda)}{g_{\mathcal{U}}(\Delta^2+2)}} = \frac{g_{\mathcal{U}}(\Delta^2 + 2)}{\lambda(\Delta^2 + 2\lambda)}. \tag{11}$$

Thus, using (5) and Lemmas 1 and 2 we get:

$$\mathbb{E}(L) \leq \frac{g_{\mathcal{U}}(\Delta^2 + 2)}{\lambda(\Delta^2 + 2\lambda)}$$

□

This theorem shows that the performance of HIPER only very weakly depends on the gap Δ between honest and malicious nodes. In addition, it is optimal up to a polynomial factor.

5 Modelling as a partially observable Markov decision Process

A Partially Observable Markov Decision Process (POMDP) [18] is a generalisation of a Markov Decision Process (MDP). More precisely, a POMDP models the relationship between an agent and its environment when the agent cannot directly observe the underlying state. A POMDP can be described as a tuple $< S, A, O, T, \Omega, R >$ where S is a finite set of states, A is a set of possible actions, O is a set of possible observations, T is a set of conditional transition probabilities and Ω is a set of conditional observation probabilities and $R : A, S \rightarrow \mathbb{R}$.

5.1 Intrusion Response and POMDP

We can model our intrusion response problem as a POMDP if we consider that a node of the network at each time-step t has a state $s_t \in S$ with $s_t = (v_t, c_t)$ where $v_t \in \{0, 1\}$ and $c_t \in \{0, 1\}$ such that:

$$v_t = \begin{cases} 0, & \text{if the node is honest,} \\ 1, & \text{if the node is malicious.} \end{cases}$$

$$c_t = \begin{cases} 0, & \text{if the node is in the network,} \\ 1, & \text{if the node is out of the network.} \end{cases}$$

where it holds that $\mathbb{P}(v_{t+1} = v_t) = 1$ since v_t is stationary (i.e. a malicious node is always malicious and an honest node remains honest) based on Assumption 1.

Additionally, at each time-step t, $\mathcal{E}$ can perform an action $a_t \in \{0, 1\}$ such that:

$$a_t = \begin{cases} 0, & \text{if } \mathcal{E} \text{ keeps the node in the network,} \\ 1, & \text{if } \mathcal{E} \text{ removes the node from the network.} \end{cases}$$

Furthermore, the following independence condition holds :

$$\mathbb{P}(v_{t+1} \mid v_t, c_t, a_t) = \mathbb{P}(v_{t+1} \mid v_t), \tag{12}$$

since the type of a node (i.e. malicious or honest) does not depend on $\mathcal{E}$'s action (i.e. remove from the network or not) neither on whether the node is in the network or out. In addition, since the type of a node never changes, it holds:

$$\mathbb{P}(v_{t+1} = j \mid v_t = j) = 1. \tag{13}$$

Consequently, we remove the time subscript from v in the sequel. On the other hand the probability that a node will be in the network depends on if it is already in or out and the action that $\mathcal{E}$ will take:

$$\mathbb{P}(c_{t+1} \mid c_t, v, a_t) = \mathbb{P}(c_{t+1} \mid c_t, a_t) \tag{14}$$

From equations (13) and (14), it is evident that the POMDP under consideration is factored.

To fully specify the model we must assume some probability distribution for the observations. Specifically, we model x_t as drawn from a Bernoulli distribution[4] with parameters u and q for honest and malicious nodes respectively.

$$\mathbb{P}(x_t = 1 \mid v = 0) = u \quad \textit{and} \quad \mathbb{P}(x_t = 1 \mid v = 1) = q. \tag{15}$$

Let $\mathbf{x}_t \triangleq (x_1, \ldots, x_t)$ be a t-length sequence of observations. From Bayes' theorem, we obtain an expression for our *belief* at time t:

$$\mathbb{P}(v = j \mid \mathbf{x}_t) = \frac{\mathbb{P}(\mathbf{x}_t \mid v = j)\,\mathbb{P}(v = j)}{\sum_{i=0}^{1} \mathbb{P}(\mathbf{x}_t \mid v = i)\,\mathbb{P}(v = i)} \tag{16}$$

where $j \in \{0, 1\}$. Thus, the expected gain at time t if $\mathcal{E}$ decides to keep a node in the network is:

$$\mathbb{E}[G_t \mid c_t = 0, \mathbf{x}_t] = \mathbb{P}(v = 0 \mid \mathbf{x}_t) \cdot g_{\mathcal{U}} - \mathbb{P}(v = 1 \mid \mathbf{x}_t) \cdot \ell_{\mathcal{Q}},$$

while the expected gain if $\mathcal{E}$ decides to remove the node from the network is always:

$$\mathbb{E}[G_t \mid c_t = 1] = 0.$$

The problem is to find a policy $\pi : X^* \to A$, mapping from the set of all possible sequences of observations to actions, maximising the total expected gain:

$$\mathbb{E}_\pi(G) = \mathbb{E}_\pi \left(\sum_{t=1}^{\infty} G_t \right). \tag{17}$$

Since future gains depend on any future observations we might obtain, the exact calculation requires enumerating all possible future observations. Consequently, the exact solution to the problem is intractable [8,11,10]. In the next section we describe possible approximations to this problem.

5.2 POMDP algorithms

We consider three algorithms: a) A *myopic* algorithm, which only considers the expected gain at the current time-step; b) An *optimistic* algorithm, which computes an upper bound on the total expected gain; c) A *finite lookahead* algorithm, which performs complete planning up to some fixed finite depth. While these algorithms have appeared before in the general MDP literature, they have not been applied before to intrusion response problems. We do not consider the most likely state approximation (MLS), since in our case there are only two possible hidden states for a node, thus, rendering the approximation far too coarse for it to be effective.

[4] This distribution is particularly convenient for computational reasons, because closed-form Bayesian inference can be performed via the Beta conjugate prior [8]. However, in principle it can be replaced with any other distribution family, without affecting the overall formalism.

Myopic. In this case, $\mathcal{E}$ only considers the expected gain for the next time-step when taking a decision. Consequently, $\mathcal{E}$ keeps the node in the network if the following condition holds:

$$\mathbb{E}[G_t \mid a_t = 0] > \mathbb{E}[G_t \mid a_t = 1]. \tag{18}$$

This algorithm is the closest to the MLS approximation among the ones considered. In fact, it is easy to see that it would be identical to MLS, as well as to a sequential probability ratio test, when $\ell_{\mathcal{Q}} = g_{\mathcal{U}}$.

Optimistic. This rule constructs an upper bound on the value of the decision to keep a node in the network, which is based on Proposition 1 in [10]. Informally, this is done by assuming that the true type of the node will be revealed at the next time-step. Then $\mathcal{E}$ keeps the node in the network if and only if:

$$\mathbb{P}(v_t = 0 \mid \mathbf{x}_t) \cdot g_{\mathcal{U}}/\lambda > \mathbb{P}(v_t = 1 \mid \mathbf{x}_t) \cdot \ell_{\mathcal{Q}}. \tag{19}$$

Intuitively, if the node is revealed to be malicious, then we can remove it at the next step and consequently we only lose $\ell_{\mathcal{Q}}$. In the converse case, we can keep it for an expected $1/\lambda$ steps.

Finite lookahead. The finite lookahead algorithm performs backwards induction [8] up to some finite depth T, at every time-step. More precisely, any sequence of observations $\mathbf{x}_t = (x_1, \ldots, x_t)$ results in a posterior probability $\mathbb{P}(v_t \mid \mathbf{x}_t)$. Let:

$$V_t \triangleq \sum_{k=t}^{\infty} G_k \tag{20}$$

be the total gain starting from time-step t. Then, the expected gain under the optimal policy is determined recursively as follows:

$$\mathbb{E}(V_t \mid \mathbf{x}_t) = \max\{0, \mathbb{E}(G_t \mid \mathbf{x}_t, a_t = 0) + \mathbb{E}(V_{t+1} \mid \mathbf{x}_t)\} \tag{21}$$

$$\mathbb{E}(V_{t+1} \mid \mathbf{x}_t) = p_t\, \mathbb{E}(G_t \mid \mathbf{x}_t, x_{t+1} = 1) + (1 - p_t)\, \mathbb{E}(V_{t+1} \mid \mathbf{x}_t, x_{t+1} = 0) \tag{22}$$

where $p_t \triangleq \mathbb{P}(x_{t+1} = 1 \mid \mathbf{x}_t) = \sum_{i=0}^{1} \mathbb{P}(x_{t+1} = 1 \mid v = i)\, \mathbb{P}(v = i \mid \mathbf{x}_t)$ is the marginal posterior probability that $x_{t+1} = 1$. For more details on this backwards induction algorithm, the reader is urged to consult [8,11].

6 Experimental Evaluation

We perform three sets of experiments. The first set investigates the performance of HIPER with various choices of the parameter δ, including the optimal choice suggested by Theorem 1. The second set compares HIPER with the *myopic* and

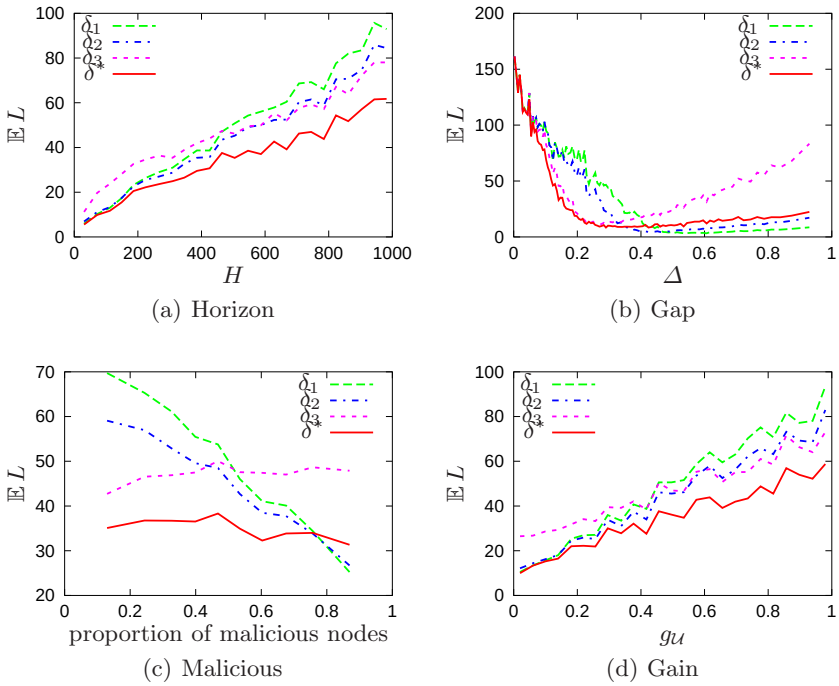
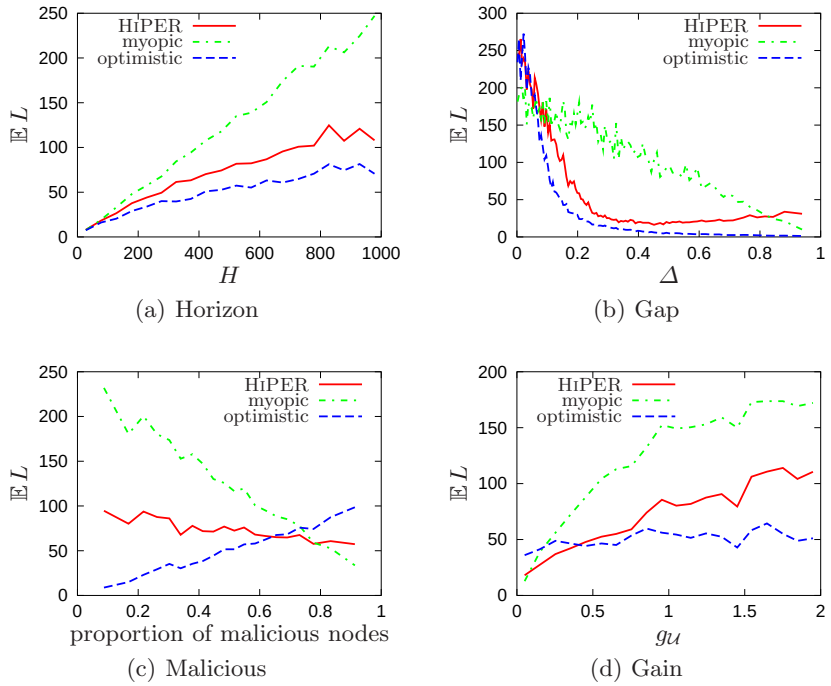


Fig. 1. Simulations with Alg. 1, for four different choices of δ. In particular $\delta_1 = 0.9$, $\delta_2 = 0.95$, $\delta_3 = 0.99$ and δ^* is chosen according to Theorem 1. It can be seen that, while the algorithm is not extremely sensitive to the exact choice of δ, the optimal value is generally more robust.

optimistic approximations. In the final set of experiments, we compare the *optimistic* with the *finite lookahead* approximation. In all cases, we collected results from 10^4 runs, with 100 nodes per run, and we plot a moving average of the *expected loss* as various network parameters change. Specifically, the first results we report (i.e. Fig. 1) are made through 10^4 experiments. For each experiment, we selected a horizon $H \sim Uniform([10, 1000])$, user and adversary parameters $u, q \sim Uniform([0, 1])$, and user gain $g_{\mathcal{U}} \sim Uniform([0, 1])$ and we set $\ell_{\mathcal{Q}} = 1$. Each experiment measured the loss for a network containing 100 nodes, each of which had a probability p of being malicious, with $p \sim Beta(2, 2)$ for each experiment. During each run, the i-th node generates a sequence of observations $x_{i,t}$ drawn from a Bernoulli distribution with parameter u if the node is honest and q if the node is malicious. The results are shown in Fig. 1 show a summary of the results, averaged over these trials. It can be seen that, while HIPER's performance is relatively robust to the choice of δ, nevertheless the optimal choice suggested by Theorem 1 generally leads to small losses.

For our second set of experiments, shown in Figure 2, we compare HIPER with the *optimistic* and *myopic* algorithms. We increased the range of user gains



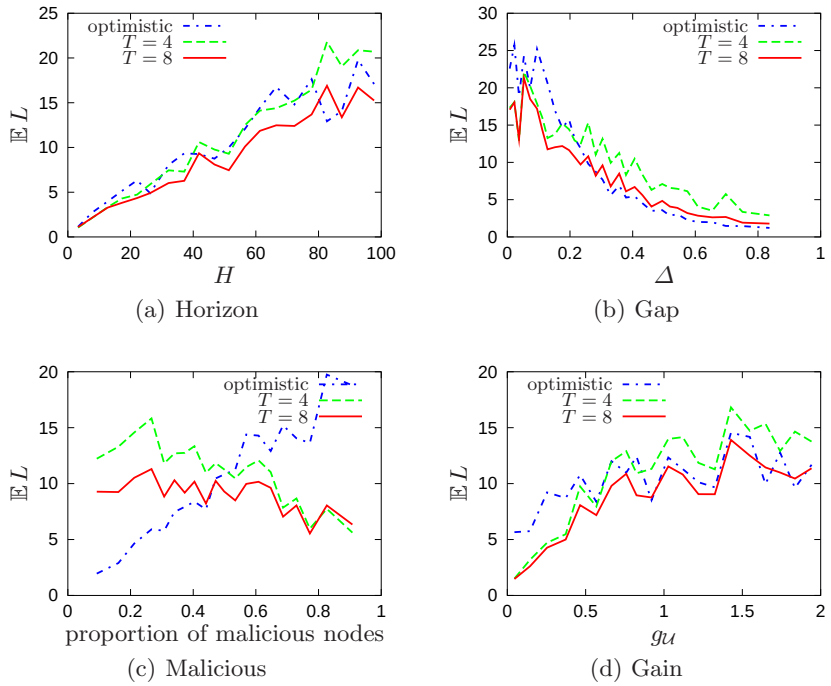
(a) Horizon (b) Gap

(c) Malicious (d) Gain

Fig. 2. Comparison of HiPER with the *myopic* solver and the *optimistic* approximation for various network conditions. It can be clearly seen that the *myopic* approximation is significantly worse than both approaches. However, the *optimistic* approach outperforms the worst-case HiPER algorithm when the proportion of malicious nodes is low. The *optimistic* approach is also better when the payment for honest nodes is high.

to $g_{\mathcal{U}} \sim Uniform([0, 2])$ compared to the previous setup, but the other experimental parameters remain the same. It is clear that the *myopic* approximation has almost always a higher loss compared to both HiPER and the *optimistic* algorithm. The latter, while performing at a similar level to HiPER, has an advantage when either the proportion of malicious is small or when $g_{\mathcal{U}}$ is large. This makes sense intuitively, since in those cases the optimism is justified. In the converse case, however, the *optimistic* approach performs worse than HiPER, which is less sensitive to the proportion of malicious nodes, since it is a worst-case approach.

Finally, we performed some experiments comparing the *optimistic* approximation with the *finite-lookahead* POMDP solvers for lookahead for T time-steps where $T \in \{4, 8\}$. While these do not solve the problem to the end of the horizon H, they plan ahead for T steps at every time-step of the simulation. Unfortunately, the complexity of these solvers is exponential in T, which limited the amount of simulations we could perform to 10^3 and we only considered horizons $H \sim Uniform([1, 100])$. These experiments are shown in Fig. 3. In compari-



(a) Horizon (b) Gap

(c) Malicious (d) Gain

Fig. 3. Comparison of the optimistic approximation with approximate *non-myopic* POMDP solvers for planning lookahead of T time-steps where $T \in \{4, 8\}$. It can be seen that, for short horizons, these perform just as well and that they are more robust to the proportion of malicious nodes in the network. However, these methods are computationally more intensive, with complexity $O(e^T)$.

son with Fig. 2, the *finite lookahead* algorithms performs much better than the *myopic* approximation and indeed the 8-step lookahead manages to slightly outperform the *optimistic* approximation. In addition, it is much more robust to the proportion of malicious nodes in the network. However, the relative advantage of the 8-step to the 4-step lookahead is relatively small for the amount of extra computation required.[5]

7 Conclusion

This paper defined a network management problem that arises frequently in communication networks. Namely, whether to remove a suspicious node from the network, with the amount of available evidence, or to collect some further data before taking the final decision. This is in fact a type of stopping problem, which we believe is of relevance to many networking applications where blacklisting may be performed. This includes applications such as automated intrusion

[5] The computational effort is exponential in T.

response, as well as ensuring fairness in peer-to-peer networks, such as [19]. To this end, we proposed and analysed, both theoretically and experimentally, a simple algorithm, HIPER, that achieves low *worst-case expected loss* relative to an oracle that knows *a priori* the type (honest or malicious) of every node in the network. In addition, we derived and compared a number of algorithms by modelling the problem as a POMDP: a *myopic* and an *optimistic* approximation, as well as a *finite lookahead* solver. Of those, the *optimistic* approximation and the partial *finite lookahead* solvers perform the best, with the *finite lookahead* methods being the most robust, while simultaneously being computationally demanding.

The main advantage of HIPER are its simplicity and lack of stringent assumptions on the distribution. This makes it suitable for deployment in most situations. However, in certain cases a full probabilistic model and computational resources are available, in which case one of the approximate solvers would be useful. The *myopic* approximation, which is almost equivalent to the widely-used MLS approximation, performs the worse. The overall best performance is offered by the *finite lookahead*. To our knowledge, neither the *optimistic* approximation, nor the *finite lookahead* methods have been applied before to this problem or more generally to intrusion response problems. They should be more generally applicable for other types of intrusion response and network management problems. It is our view that they are inherently more suitable than other approximations such as the most likely state (MLS) approximation (or equivalently, a sequential probability ratio test) which in our setting produces an essentially random policy.

For future work, we would like to extend our theoretical analysis to the performance of the *optimistic* and the *finite lookahead* algorithms. In addition, it would be interesting to examine a more general game-theoretic scenarios, including strategic attackers[2,1], as well as colluding nodes. Finally, we would like to generalise our setting so that observations must be *explicitly* gathered from each node, where it is not possible to continuously sample all nodes due to budget constraints. In fact, the sampling problem in the context of intrusion detection, has been recently studied by [14,4]. A natural extension of our work would consequently be to optimally combine sampling and response policies.

A Proofs

This section collects the missing proofs from the main text.

Proof ((Lemma 1)). Since the node i under consideration is malicious, i.e. $i \in \mathcal{Q}$, it holds that: $\mathbb{E}[x_{i,t} \mid \mathcal{Q}] = q$. Then, we have:

$$\mathbb{E}[\theta_t \mid \mathcal{Q}] = \mathbb{E}\left[\frac{1}{t} \cdot \sum_{k=1}^{t} x_{i,k} \,\middle|\, \mathcal{Q}\right] = \frac{1}{t} \sum_{k=1}^{t} \mathbb{E}[x_{i,k} \mid \mathcal{Q}] = \frac{1}{t} \cdot t \cdot q = q.$$

From Hoeffding's inequality (*Lemma* 3, in the Appendix), we have:

$$\mathbb{P}\left(|\theta_t - q| > \epsilon_t \mid \mathcal{Q}\right) \leq 2\exp(-2t\epsilon_t^2), \tag{23}$$

where $\epsilon_t > 0$ and $\mathbb{P}\left(|\theta_t - q| > \epsilon_t \mid \mathcal{Q}\right)$ denotes the probability that θ_t (which is random) is very far away from q (which is fixed).
Now let set:

$$\epsilon_t = \sqrt{\frac{\ln(2/\delta)}{2t}}$$

as in Algorithm 1. Then, since equation 23 holds for any $\epsilon_t > 0$, we get that the probability of keeping a malicious node $i \in \mathcal{Q}$ in the network is at most δ:

$$\mathbb{P}\left(|\theta_t - q| > \sqrt{\frac{\ln(2/\delta)}{2t}} \mid \mathcal{Q}\right) < \delta \tag{24}$$

Thus, we have:

$$\mathbb{E}[L \mid \mathcal{Q}] = \mathbb{E}[N \mid \mathcal{Q}] \cdot \ell_{\mathcal{Q}} = \sum_{t=0}^{\infty} \mathbb{P}(N = t \mid \mathcal{Q}) \cdot t \cdot \ell_{\mathcal{Q}}$$

Thus:

$$\begin{aligned}\mathbb{E}[L \mid \mathcal{Q}] = \ell_{\mathcal{Q}} \sum_{t=0}^{\infty} \mathbb{P}(N = t \mid \mathcal{Q}) \cdot t &\leq \ell_{\mathcal{Q}} \sum_{t=0}^{\infty} \delta^{t-1} \cdot t \\ &= \frac{\ell_{\mathcal{Q}}}{(1-\delta)^2}\end{aligned}$$

□

Proof ((Lemma 2)). We denote by N the time-step at which $\mathcal{E}$ removes node i from the network. Then, the function $g : \mathbb{N}^2 \to \mathbb{R}$ that gives us the gain for each node i is defined as:

$$g(n, h) \triangleq \min\{n, h\} \cdot g_{\mathcal{U}} \tag{25}$$

where $h \in H$ and $n \in N$.
Since the node i under consideration is honest, i.e. $i \in \mathcal{U}$, we have $\mathbb{E}[x_{i,t} \mid \mathcal{U}] = u$. Without loss of generality we assume that: $u = q + \Delta$, where $\Delta > 0$. So we only need $\mathbb{P}(\theta_t - q < \epsilon_t \mid \mathcal{U})$.
Since $q = u - \Delta$ from the Hoeffding inequality (*Lemma* 3, in the Appendix), we have:

$$\begin{aligned}\mathbb{P}(N = t \mid \mathcal{U}) \leq \mathbb{P}(N \leq t) &\leq \mathbb{P}(\theta_t - u < \epsilon_t - \Delta \mid \mathcal{U}) \\ &\leq \exp(-2 \cdot t(\epsilon_t - \Delta)^2)\end{aligned}$$

where $\Delta - \epsilon_t > 0$. It holds that:

$$\begin{aligned}&\mathbb{E}[G \mid \mathcal{U}, N = n] = \\ &\sum_{n=0}^{\infty} \mathbb{P}(H = h \mid \mathcal{U}, N = n)\, \mathbb{E}[G \mid \mathcal{U}, N = n, H = h]\end{aligned} \tag{26}$$

But it holds that:

$$\mathbb{E}[G \mid \mathcal{U}, N = n, H] = g(n, h)$$

and since $h \in H$ and $n \in N$ are independent we have:

$$\mathbb{P}(H = h \mid \mathcal{U}, N = n) = \mathbb{P}(H = h \mid \mathcal{U}).$$

Thus,

$$\begin{aligned}
&\mathbb{E}[G \mid \mathcal{U}, N = n] = \\
&\sum_{h=0}^{\infty} \mathbb{P}(H = h \mid \mathcal{U}) \cdot g(n, h) = \\
&\sum_{h=0} \mathbb{P}(H = h \mid \mathcal{U}) \min\{n, h\} \cdot g_{\mathcal{U}} = \\
&g_{\mathcal{U}} \cdot \Big\{ \sum_{h=0}^{n-1} \mathbb{P}(H = h \mid \mathcal{U}) \cdot h + \sum_{h=n}^{\infty} \mathbb{P}(H = h \mid \mathcal{U}) \cdot n \Big\}
\end{aligned} \tag{27}$$

The expected loss is given by subtracting from the expected gain of the oracle policy, when $\mathcal{E}$ never removes the node from the network (i.e. $N = \infty$), the expected gain when $\mathcal{E}$ removes the node at the time-step $N = n$. Thus, it holds:

$$\begin{aligned}
&\mathbb{E}[L \mid \mathcal{U}, N = n] = \\
&\mathbb{E}[G \mid \mathcal{U}, N = \infty] - \mathbb{E}[G \mid \mathcal{U}, N = n] = \\
&\lim_{n \to \infty} (\mathbb{E}[G \mid \mathcal{U}, N = n]) - \mathbb{E}[G \mid \mathcal{U}, N = n] = \\
&g_{\mathcal{U}} \sum_{h=0}^{\infty} \mathbb{P}(H = h) \cdot h - g_{\mathcal{U}} \Big\{ \sum_{h=0}^{n-1} \mathbb{P}(H = h) \cdot h + \sum_{h=n}^{\infty} \mathbb{P}(H = h) \cdot n \Big\} \\
&= g_{\mathcal{U}} \Big\{ \sum_{h=n}^{\infty} \mathbb{P}(H = h) \cdot h - \sum_{h=n}^{\infty} \mathbb{P}(H = h) \cdot n \Big\}
\end{aligned} \tag{28}$$

Since, by definition $\mathbb{P}(H = h+1 \mid H > h) = \lambda$, we have $\mathbb{P}(H = h) = (1-\lambda)^{h-1}\lambda$. Consequently,

$$\begin{aligned}
\mathbb{E}[L \mid \mathcal{U}, N = n] =& g_{\mathcal{U}} \cdot \lambda \Big(\sum_{h=n}^{\infty} (1-\lambda)^{h-1} \cdot h - \sum_{h=n}^{\infty} (1-\lambda)^{h-1} \cdot n \Big) \\
=& g_{\mathcal{U}} \cdot \frac{(1-\lambda)^n}{\lambda}
\end{aligned} \tag{29}$$

Thus, we have:

$$\begin{aligned}
\mathbb{E}[L \mid \mathcal{U}] &= \sum_{t=0}^{\infty} \mathbb{P}(N = t \mid \mathcal{U}) \, \mathbb{E}[L | N = t] \\
&\leq \sum_{t=0}^{\infty} \exp(-2 \cdot t \cdot (\epsilon_t - \Delta)^2) \cdot g_{\mathcal{U}} \frac{(1-\lambda)^t}{\lambda}
\end{aligned}$$

Since the algorithm uses $\epsilon_t = \frac{\Delta}{\sqrt{t}}$, we have:

$$\begin{aligned}
E[L \mid \mathcal{U}] \leq & \frac{g_{\mathcal{U}}}{\lambda} \sum_{t=0}^{\infty} \exp\left(-2 \cdot t \left[\frac{\Delta}{\sqrt{t}} - \Delta\right]^2\right) (1-\lambda)^t \\
= & \frac{g_{\mathcal{U}}}{\lambda} \sum_{t=0}^{\infty} \exp\left(-2\Delta^2(\sqrt{t}-1)^2\right) (1-\lambda)^t \\
\leq & \frac{g_{\mathcal{U}}}{\lambda} \sum_{t=0}^{\infty} \exp\left(-2\Delta^2 \left[\sqrt{t} - \sqrt{\frac{t}{2}}\right]^2\right) (1-\lambda)^t \\
= & \frac{g_{\mathcal{U}}}{\lambda} \sum_{t=0}^{\infty} \left[\exp\left(-\frac{\Delta^2}{2}\right)(1-\lambda)\right]^t \\
= & \frac{g_{\mathcal{U}}}{\left[1 - \exp\left(-\frac{\Delta^2}{2}\right)(1-\lambda)\right]\lambda} \\
\leq & \frac{g_{\mathcal{U}}(\Delta^2+2)}{\lambda(\Delta^2+2\lambda)} \qquad (30)
\end{aligned}$$

where $t \geq 2$. □

B Additional results

Definition 1 (Bernoulli distribution). *If $X_1, \ldots, X_n$ are independent Bernoulli random variables with $X_k \in \{0,1\}$ and $\mathbb{P}(X_k = 1) = \mu$ for all k, then*

$$\mathbb{P}\left(\sum_{k=1}^{n} X_k \geq u\right) = \sum_{k=0}^{u} \binom{n}{k} \mu^k (1-\mu)^{n-k}. \qquad (31)$$

Lemma 3 (Hoeffding). *For independent random variables $X_1, \ldots, X_n$ such that $X_i \in [a_i, b_i]$, with $\mu_i \triangleq \mathbb{E}\, X_i$ and $t > 0$:*

$$\mathbb{P}\left(\sum_{i=1}^{n} X_i \geq \sum_{i=1}^{n} \mu_i + nt\right) \leq \exp\left(-\frac{2n^2t^2}{\sum_{i=1}^{n}(b_i - a_i)^2}\right).$$

The same in equality holds for $\sum_{i=1}^{n} X_i \leq \sum_{i=1}^{n} \mu_i - nt$.

References

1.
2. N. Bao, P. Kreidl, and J. Musacchio. A network security classification game. In *GameNets 2011*, 2011.
3. C. Boutilier. A pomdp formulation of preference elicitation problems. In *Proceedings of the National Conference on Artificial Intelligence*, pages 239–246. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2002.

4. S. Bu, F. Yu, X. Liu, and H. Tang. Structural results for combined continuous user authentication and intrusion detection in high security mobile ad-hoc networks. *Wireless Communications, IEEE Transactions on*, (99):1–10, 2011.
5. T. Bui, M. Poel, A. Nijholt, and J. Zwiers. A tractable ddn-pomdp approach to affective dialogue modeling for general probabilistic frame-based dialogue systems. 2006.
6. A. Cassandra. *Exact and Approximate Algorithms for Partially Observable Markov Decision Processes*. PhD thesis, Brown University, 1998.
7. N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning and Games*. 2006.
8. M. DeGroot. *Optimal Statistical Decisions*. John Wiley & Sons, 1970. Republished in 2004.
9. S. Dejmal, A. Fern, and T. Nguyen. Reinforcement learning for vulnerability assessment in peer-to-peer networks. In *Proceedings of the 20th national conference on Innovative applications of artificial intelligence*, pages 1655–1662, 2008.
10. C. Dimitrakakis. Complexity of stochastic branch and bound methods for belief tree search in Bayesian reinforcement learning. In *2nd international conference on agents and artificial intelligence (ICAART 2010)*, pages 259–264, Valencia, Spain, 2009. ISNTICC, Springer.
11. M. O. Duff. *Optimal Learning Computational Procedures for Bayes-adaptive Markov Decision Processes*. PhD thesis, University of Massachusetts at Amherst, 2002.
12. Y. He and K. Chong. Sensor scheduling for target tracking in sensor networks. In *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, volume 1, pages 743–748. IEEE, 2004.
13. W. Lee, W. Fan, M. Millerand, S. Stolfo, and E. Zadok. Toward Cost-Sensitive Modeling for Intrusion Detection and Response. *Journal of computer Security*, 10:5–22, 2000.
14. K. Liu and Q. Zhao. Dynamic intrusion detection in resource-constrained cyber networks. Technical Report arXiv:112.0101, arXiv, 2011.
15. S. Ross, J. Pineau, S. Paquet, and B. Chaib-draa. Online planning algorithms for POMDPs. *Journal of Artificial Intelligence Resesarch*, 32:663–704, July 2008.
16. Z. Saigol and U. of Birmingham. School of Computer Science. *Information-lookahead planning for AUV mapping*. School of Computer Science, University of Birmingham, 2009.
17. P. Si, F. Yu, H. Ji, and V. Leung. Distributed sender scheduling for multimedia transmission in wireless mobile peer-to-peer networks. *Wireless Communications, IEEE Transactions on*, 8(9):4594–4603, 2009.
18. R. Smallwood and E. Sondik. The Optimal Control of Partially Observable Markov Processes over a Finite Horizon. *Operational Research*, 21:1071–88, 1973.
19. A. Vieira, S. Campos, and J. Almeida. Fighting attacks in p2p live streaming. simpler is better. In *INFOCOM Workshops 2009, IEEE*, pages 1–2. IEEE, 2009.
20. X. Zan, F. Gao, J. Han, X. Liu, and J. Zhou. A Hierarchical and Factored POMDP based Automated Intrusion Response Framework. In *Proceedings of the 2nd International Conference on Software Technology and Engineering (ICSTE)*, volume 2, pages 410–414. IEEE, 2010.
21. Z. Zhang, P.-H. Ho, and L. He. Measuring IDS-estimated Attack Impacts for Rational Incident Response: A Decision Theoretic Approach. *Computers & Security*, 28:605–614, 2009.
22. S. Zonouz, H. Khurana, W. Sanders, and Y. T.M. RRE: A Game-Theoretic Intrusion Response and Recovery Engine. In *Proceedings of the IEEE/IFIP Inter-*

national Conference on Dependable Systems & Networks, 2009 (DSN'09), pages 439–448, Lisbon, Portugal, 29 June – 2 July 2009.