Optimizing the Decision to Expel Attackers from an Information System

Ning Bao and John Musacchio

***Abstract*— The conventional reaction after detecting an attacker in an information system is to expel the attacker immediately. However the attacker is likely to attempt to reenter the system, and if the attacker succeeds in reentering, it might take some time for the defender's Intrusion Detection System (IDS) to re-detect the attacker's presence. In this interaction, both the attacker and defender are learning about each other – their vulnerabilities, intentions, and methods. Moreover, during periods when the attacker has reentered the system undetected, he is likely learning faster than the defender. The more the attacker learns, the greater the chance that he succeeds in his objective – whether it be stealing information, inserting malware, or some other objective. Conversely, the greater the defender's knowledge, the more likely that the defender can prevent the attacker from succeeding. In this setting, we study the defender's optimal strategy for expelling or not expelling an intruder. We find that the policy of always expelling the attacker can be far from optimal. Furthermore, by formulating the problem as a Markov decision process (MDP), we find how the optimal decision depends on the state variables and model parameters that characterize the IDS's detection rate and the attacker's persistence.**

I. Introduction

The conventional approach to cyber defense against attackers could be characterized as: (1) make the system difficult to infiltrate; (2) detect the infiltrations as soon as possible; (3) expel the attackers when their existence are detected. A fundamental problem with this approach is that it does not consider the reality that attackers are likely to try again after failures. Arguably, some intrusion detection technologies with adaptive learning capabilities may be able to detect attackers with increasing efficiency with each successive intrusion attempt from a given attacker. However, when attackers do succeed in infiltrating a system, it is possible that they might learn more quickly about the defender's defense mechanisms than the defender learns about the attacker. Consequently, each attempt of attack is more and more likely to succeed.

During the course of an attack, the defender may choose either to expel the attacker once he detects his presence, or to keep his in the system in order to observe and learn about the attacker. If the defender could "out-learn" the attacker, i.e. learn about the attacker faster than he learns about the defender, with the help of that intelligence the defender may be able to totally thwart the attacker's infiltration and ensure the security of the system against this attacker in the long run.

This research is supported by Air Force Office of Scientific Research (AFOSR) under grant AFOSR FA9550-09-1-0049.

N. Bao and J. Musacchio are with the Technology of Information Management Program, University of California, Santa Cruz, Santa Cruz, CA 95064, USA {nbao, johnm}@soe.ucsc.edu

In this paper, we study the defender strategy (policy) optimization problem in the presence of this *learning effect*. We formulate the problem as a single-controller Markov decision process (MDP), and investigate the optimal defender policies under various conditions.

A. Related Work

One of the best ways the research community learns about attackers and their techniques are through the use of honeynets, and the HoneyNet project is one of the most important projects advancing this approach [1]. The honeynet is a sandbox environment with no other purpose than to attract attackers and record data about their attack techniques. Our work considers the potential for this kind of learning in an actual in-use information system. There of course is a trade-off between the opportunity to learn about the attacker and the risk that attacker does damage while the defender tries to learn. It is this trade-off that we explicitly model in this work.

In our model, the system is abstracted as a relatively simple Markov decision process. Some authors also investigate finer-grained abstractions, where an attack is decomposed into several states and transitions are driven by atomic attack actions [2], [3], [4], [5]. For instance, Jha *et. al.* [5] proposed the framework of attack graph and employed MDP techniques to compute the probabilities of attacker success. However, none of these work considers the possibility that the defender may learn and counteract the attacks.

A growing body of work uses game theory to investigate computer security [6], [7]. In a series papers [8], [9], [10], Alpcan and Başar proposed a game-theoretic framework to model the interaction between the Intrusion Detection System (IDS) and attackers. Though in our current formulation, only one player has a non-trivial strategy choice – making it an MDP rather than a game – in our future work we will investigate models with a richer strategy space for both attackers and defenders.

The rest of the paper is organized as follows: Section II describes our proposed model in detail. The analysis of the model is given in Section III and numerical studies are discussed in Section IV. At last, Section V summarizes our work and identifies future research directions.

II. Model

We use a simple discrete-time MDP to model the system. Our model proceeds in discrete time slots indexed by k. At any time k, the state of the system is described by four state variables. The state variable $c_k \in \{C, NC\}$ describes whether the attacker is "Connected" to the secured information system or "Not Connected". Similarly, $d_k \in \{D, ND\}$

indicates whether the defender has either "Detected" or "Not Detected" the attacker's connection. The other two variables, $x_k, y_k \in [0,1]$, is used to represent the *knowledge* that the attacker and defender have at time k, respectively.

In our context, the *knowledge* of the attacker denotes any knowledge that may help the attacker to achieve his goal. This can include general information such as the network topology of defender's information system, the versions of operating systems and applications running on the servers; or more advanced intelligence such as the vulnerabilities in the defender's operating system or application, the structure of the file system of a particular file server. For a defender, *knowledge* refers to any information could aid the defender counteract the attacker's current and future attacks. This may include, for instance, the attacker's objectives, methods used, estimated technical level, etc. The evolution of *knowledge* then indicates the process of *learning*.

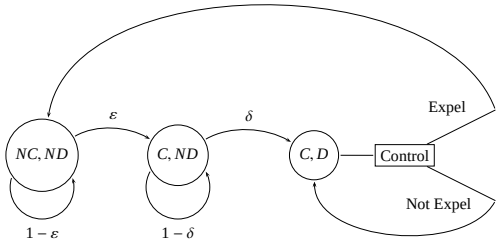

Figure 1: The dynamics of the connection and detection processes.

The system evolves according to the rules described below, and the connection and detection aspects is illustrated in Figure 1.

- Each period that the attacker is disconnected from the defender's system, he may (re-)connect with probability $p_{\text{connect}} = \varepsilon$.
- Each period that the attacker is connected but not yet detected, the defender may detect the existence of attacker with probability $p_{\text{detect}} = \delta$. This probability reflects the capability of the Intrusion Detection System (IDS) deployed by the defender.
- After connection (in state $(C, \cdot)$), the attacker could achieve his objective with probability $p_{\text{success}}(k) = \gamma x_k(1 - y_k)$ where $\gamma \in (0,1]$ is a scalar. Note that p_{success} increases as the attacker's knowledge x_k grows, and decreases as the defender's knowledge y_k grows. We refer to this formulation as the *single goal* formulation. In an alternative formulation, we consider the case where the attacker may have multiple goals during the course of an attack and each goal will yield 1 unit of reward to the attacker, the quantity $\gamma x_k(1-y_k)$ is then the expected reward for the attacker in each period; and we call it *multiple goals* formulation.
- During periods that the attacker is connected (in state $(C, \cdot)$), he could gather information about the defender's system. The defender may also learn about the attacker during periods that the defender knows the presence (in state (C, D)) of the attacker. To make the model tractable, we use two simple types of learning curves to model the knowledge increase – geometric and linear learning curves. For the *geometric case*, x_k, y_k evolves according to the following recursive expressions during an learning period:

$$x_k = x_{k-1} + \alpha(1 - x_{k-1}),$$
$$y_k = y_{k-1} + \beta(1 - y_{k-1}),$$

where $\alpha, \beta \in (0,1)$ are the corresponding learning parameters represents the speed of learning of the attacker and defender, respectively. We also consider the *linear case* which facilitates the analysis of the corresponding MDP problem as to make the state space finite:

$$x_k = \begin{cases} x_{k-1} + \alpha & \text{if } 0 \le k \le \lfloor 1/\alpha \rfloor \\ 1 & \text{otherwise} \end{cases},$$
$$y_k = \begin{cases} y_{k-1} + \beta & \text{if } 0 \le k \le \lfloor 1/\beta \rfloor \\ 1 & \text{otherwise} \end{cases},$$

where $\alpha, \beta \in (0,1)$ are the slopes of the corresponding learning curves.
- Every period that the attacker is connected and detected, the defender has a control decision to expel the attacker or not. Expelling the attacker will drive the system to state (NC, ND); otherwise, the system stays at (C, D) for one period. This is the only state where the defender has the opportunity to apply control, and the attacker has no control choice in this model.

III. Analysis

The model we have developed in Section II is stochastic, and features a decision to be made by one agent, the defender, in some states. Naturally, we formulate the problem as a Markov decision process (MDP). In this section, we analyze the proposed model in detail. First, we formulate the problem into an MDP. Then we analyze two MDP formulations based on different assumptions of the attacker.

A. Markov Decision Process Formulation

A Markov decision process (MDP) [11], [12] is defined by a tuple, $(\mathbf{S}, \mathbf{A}, p, r)$, where $\mathbf{S}$ is the set of *states*, $\mathbf{A}$ is the set of *actions*, $p : \mathbf{S} \times \mathbf{A} \times \mathbf{S} \mapsto [0,1]$ is the *transition* function, and $r : \mathbf{S} \times \mathbf{A} \mapsto \mathbf{R}$ is the cost function. We use $\mathbf{A}(s)$ to denote the set of *admissible* actions at state s. A discrete-time MDP proceeds in discrete time slots, $k = 0, 1, \ldots$, and the state/action/cost at time k is denoted by S_k, A_k, R_k. The transition function defines a probability distribution across states conditioned on the current state and the action taken, i.e. $p(s'|s,a) := P\{S_{k+1} = s' | S_k = s, A_k = a\}$. The immediate cost incurred at time k is given by $R_k := r(s,a)$. A *control/decision*, μ_k, made by the controller at time k is a function $\mu_k : \mathbf{S} \times \mathbf{A}(s) \mapsto [0,1]$ which defines a probability distribution across all possible actions available given the current state; and a *policy* $\pi = \{\mu_1, \mu_2, \ldots\}$ is a sequence of decisions indexed by time k.

The solution of an MDP is to find the *optimal policy* which minimizes the total expected cost. In order to make the total expected cost well defined, there are several classes of MDP

problems. Particularly, we are interested in two classes which fit our model: the Stochastic Shortest Path (SSP) problem and the discounted MDP [12].

In view of the MDP definition above, we formulate our model as an MDP by identifying the corresponding elements. For simplicity, we start with the linear case. The state space is defined as $S := X \times Y \times W$ where $X := \{0, \alpha, 2\alpha, \ldots, 1\}$, $Y := \{0, \beta, 2\beta, \ldots, 1\}$ denote the set of attacker and defender knowledge values respectively, and $W := \{(NC, ND), (C, ND), (C, D)\} = \{\mathbf{1}, \mathbf{2}, \mathbf{3}\}$ indicates the connection-detection state. Clearly, the cardinality of S is $|S| = \lceil 1/\alpha \rceil \times \lceil 1/\beta \rceil \times 3$ which is finite. The state variable is represented by a tuple $S_k := (x_k, y_k, w_k)$ in which x_k, y_k are the attacker and defender knowledge values and w_k is the connection-detection state at time k. The *planning horizon* of this MDP is infinite theoretically (Cf. Section III-B). In such *infinite horizon* problems, one usually ignores the time index in state and policy notation. Hence, we will *not* use subscript to denote time index in the sequel unless otherwise specified. Instead, state is denoted by (x, y, w) in general. Besides, we will use the notation x_m, y_n to index elements in the set X, Y in ascending order respectively (Cf. Appendix B).

One could imagine the state space as the following: for each $w \in \{(NC, ND), (C, ND), (C, D)\}$ the (x, y) pairs form a two-dimensional grid, and there are 3 such grids all together. Figure 2 visualizes the state space. In Section II

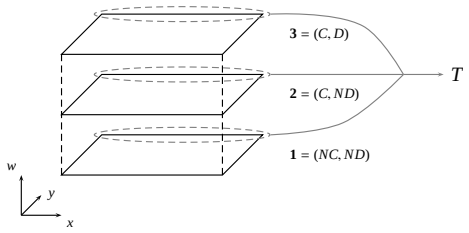

Figure 2: Illustration of the state space

we have defined the quantity $\gamma x(1 - y)$ as the probability of success of the attacker in the *single goal* formulation, or the expected reward gained by the attacker in the *multiple goals* formulation. We assume that the cost suffered by the defender also equals to this quantity. Note that once $y = 1$, meaning that the defender has "complete" knowledge about the attacker, we have $\gamma x(1-y) = 0$ and no further cost could be incurred. In this case, the defender is able to totally thwart future attacks and the process is over. This interpretation naturally leads to a set of *termination states* [12] $T := \{(x, y, w) : y = 1\}$ as illustrated in Figure 2.

The defender can only make decisions in the states $(\cdot, \cdot, \mathbf{3})$, and the action set is $\mathbf{A} = \{\text{Expel, Not-Expel}\}$. The transition probabilities are defined as the following based on the description in Section II:

$$P\{(x, y, \mathbf{2}) \mid (x, y, \mathbf{1})\} = \varepsilon, P\{(x, y, \mathbf{1}) \mid (x, y, \mathbf{1})\} = 1 - \varepsilon,$$
$$P\{(x + \alpha, y, \mathbf{3}) \mid (x, y, \mathbf{2})\} = \delta,$$
$$P\{(x + \alpha, y, \mathbf{2}) \mid (x, y, \mathbf{2})\} = 1 - \delta,$$
$$P\{(x + \alpha, y + \beta, \mathbf{1}) \mid (x, y, \mathbf{3}), \text{Expel}\} = 1,$$
$$P\{(x + \alpha, y + \beta, \mathbf{3}) \mid (x, y, \mathbf{3}), \text{Not-Expel}\} = 1,$$

where $(x+\alpha)$ is truncated to 1 if $x+\alpha \geq 1$, and similar for $(y + \beta)$ due to the definition of the linear learning curves.

Notice that in our model the immediate cost function *does not* depend on the action of the defender, therefore $r : \mathbf{S} \mapsto \mathbf{R}$ and we denote it as $r(x, y, w)$. In the single goal formulation, the measure of interest would be the cumulative probability of success, p_c. We can express p_c in terms of the per-stage probability of success as $p_c = 1 - \prod_{i=0}^{k}(1 - p_{\text{success}}(i)) = 1 - \exp\left\{\sum_{i=0}^{k} \log(1 - p_{\text{success}}(i))\right\}$. Notice that since the exponential function is monotone, maximizing some function $\exp\{f(\cdot)\}$ is equivalent to minimizing $-f(\cdot)$. Consequently, the corresponding immediate (per-stage) cost, r, of the MDP is defined as:

$$r(x, y, \mathbf{1}) = 0,$$
$$r(x, y, w) = -\log(1 - \gamma x(1 - y)) \quad \text{for } w \in \{\mathbf{2}, \mathbf{3}\}.$$

Alternatively, we define $r(x, y, w) = \gamma x(1 - y)$ for $w \in \{\mathbf{2}, \mathbf{3}\}$ for the multiple goals formulation. As a matter of fact, as long as r satisfies: (1) $r(x, y, \mathbf{1}) = 0$, (2) $r(x, 1, w) = 0$, and (3) $\partial r(x, y)/\partial x > 0, \partial r/\partial y < 0$, our analysis of the MDP model won't be affected. For notational convenience, we shall use $r(x, y) := r(x, y, w)$ for $w \in \{\mathbf{2}, \mathbf{3}\}$ and 0 in place of $r(x, y, \mathbf{1})$ directly in the sequel.

We consider two different assumptions of the attacker. In the case where the attacker is "persistent," he will relentlessly try to re-connect to the system after being kicked out; and the immediate cost is accumulated without discounting. We call this case the undiscounted MDP. On the other hand, if the attacker is "impersistent," we modify the model by introducing a fixed probability that the attacker gives up at each state. This modified model naturally leads to a MDP with discounted total cost criterion.

B. Undiscounted MDP

As discussed in Section III-A, there is a set T of natural cost-free *termination states* in which the defender has learned the attacker completely. Moreover, the process will reach those termination states regardless the policy taken by the defender. To see this, first notice that the system will jump from $(x, y, \mathbf{1})$ to $(x', y, \mathbf{3})$ with probability 1 for any $x, y \in X, Y$ and $x' > x$ due to the positive connecting and detecting probabilities. Moreover, at state $(x, y, \mathbf{3})$ the system proceeds to either $(x', y', \mathbf{1})$ or $(x', y', \mathbf{3})$ such that $x' \geq x, y' \geq y$. Even the defender chooses to Expel and the attacker is disconnected, the same argument implies that the defender's knowledge will grow again after some period of time. Hence, the defender's knowledge increases towards 1 and the system will ultimately reach the termination states. However, the number of stages taken to arrive at the termination is stochastic.

Since the system will eventually end up with some termination state with no future cost, the total expected cost is well-defined and could be directly minimized to find the optimal policy. The total expected cost starting from

state $S_0 = s$ incurred by some policy π is given by $v_\pi(s) = \sum_{k=0}^{\infty} \mathrm{E}_\pi [R_k | S_0 = s]$ and the function $v_\pi : \mathbf{S} \mapsto \mathbf{R}$ is called the *cost function* of policy π, or in vector notation $\mathbf{v}_\pi$ is called the *cost vector*. Hence, the optimal cost vector $\mathbf{v}^*$ and the corresponding optimal policy π^* is defined as $v^*(s) = v_{\pi^*}(s) = \min_\pi \{\sum_{k=0}^{\infty} \mathrm{E}_\pi [R_k | S_0 = s]\}$. One important theoretic results of SSP problem is that the *Bellman's equation* holds for the optimal cost vector [12]:

$$v^*(s) = \min_{a \in \mathbf{A}(s)} \left\{ r(s) + \sum_{s' \in \mathbf{S}} p(s'|s,a) v^*(s') \right\}.$$

Moreover, there exists an optimal *stationary deterministic* policy which specifies a single action at each state regardless of the time (stage) of the SSP problem.

The SSP problem has an intuitive interpretation as follows: Imagine a digraph with vertices corresponding to states and arcs as transitions. If we assign the immediate costs to each arc, the problem becomes finding a policy which gives the *expected shortest path* from the initial state to the set of termination states T. Consequently, a good policy shall drive the system to the termination states as soon as possible. Besides, the termination states are where $y = 1$ in our model. Hence a good policy shall make the defender learn as fast as possible. Since the Not-Expel action would give the defender the opportunity to learn again at the next stage while the Expel action would postpone the defender's learning, intuitively, the extreme policy where the defender *always* choose to retain the attacker shall be optimal and we call it *Never-Expel*. In contrast, we denote the policy which expels the attacker whenever his presence is detected as *Always-Expel*.

The above intuitive argument shows that Never-Expel policy might be the optimal policy and we prove this formally by first exploiting the structure of the optimal cost vector in Proposition 1.

Proposition 1 *For the undiscounted MDP (persistent attacker) with linear learning curves, the optimal cost vector has the following properties:*

(a) *For each $y \in Y$ and $x > x'$, $v^*(x, y, \mathbf{3}) > v^*(x', y, \mathbf{3})$;*
(b) *For each $y \in Y$ and $x > x'$, $v^*(x, y, \mathbf{2}) > v^*(x', y, \mathbf{2})$;*
(c) *For any $x \in X, y \in Y$, $v^*(x, y, \mathbf{2}) > v^*(x, y, \mathbf{3})$ and $v^*(x, y, \mathbf{1}) = v^*(x, y, \mathbf{2})$;*

where x, y denote the defender and attacker's knowledge respectively and $\{\mathbf{1}, \mathbf{2}, \mathbf{3}\}$ corresponds to the connection-detection states $\{(NC, ND), (C, ND), (C, D)\}$.

Proof: Refer to Appendix A. ■

From the Bellman's equation of our formulation, one can make the observation that the optimal control is simply determined by the comparison of the optimal costs $v^*(x, y, \mathbf{3})$ and $v^*(x, y, \mathbf{1})$. Consequently, the property (c) of Proposition 1 implies that the optimal actions shall be Not-Expel for all (x, y) pairs, and this leads to the next result.

Proposition 2 *For the undiscounted MDP (persistent attacker) with linear learning curves, the* Never-Expel *policy is optimal and dominates any other policy.*

Next, we extend the result to the geometric case and a minor embellishment of the model.

1) The Geometric Case: The linear case of the original model enabled us using the termination states as the base case of induction. However, the geometric case does not have natural termination states (since y_k only asymptotically approaches but never equals to 1 as the time index k grows). We define the state space for the geometric case as: $X := \{1-(1-\alpha)^m \ : \ m \in \mathbf{Z}^+\}$, $Y := \{1-(1-\beta)^n \ : \ n \in \mathbf{Z}^+\}$ and W is the same as before. In order to extend the above results to the geometric case, we need to establish an ϵ-argument to show that for any $\epsilon > 0, \exists N \in \mathbf{Z}^+$ such that $v_\pi(x, y_n, \cdot) < \epsilon$ holds for all $n > N$, where $x \in X, y_n = 1 - (1-\beta)^n, n \in \mathbf{Z}^+$ and π is any given policy. In other words, we may specify a set of states with arbitrarily small cost-to-go's as the termination states. Then, we effectively transform the problem into a SSP as the linear case, and the proof of Proposition 1 works for the geometric case as well. The details of the ϵ-argument is given in Appendix B.

2) Boosting Factor Upon Expulsion: A simple embellishment of the original model is to introduce a knowledge boosting factor, f. When the defender chooses to expel the attacker, the attacker's knowledge grows according to $x_k = x_{k-1} + f\alpha(1 - x_{k-1})$ (geometric case) or $x_k = x_{k-1} + f\alpha$ (linear case) where $f > 1$ and k is the time index. This expression reflects the possibility that the attacker may learn faster in an expulsion period than in a period he stays connected without expulsion. This reflects the effect that the attacker may learn something about the reason of his failure (being detected) so that he can improve tactics next time.

This embellishment shall not affect the results derived in this section because it only makes expulsion less attractive. Yet, we need to modify the state space slightly. For the linear case, we define the set $X := \{(i + jf)\alpha \ : \ i + jf \leq 1/\alpha, i, j \in \mathbf{Z}^+\} \cup \{1\}$. Notice that X is finite due to the constraint $i + jf \leq 1/\alpha$ and i, j being positive integers. For the geometric case, we define $X := \{1-(1-\alpha)^i(1-f\alpha)^j \ : \ i, j \in \mathbf{Z}^+\}$ which is still countable. Corresponding Y sets of two cases can be defined analogously. Since the extended state space retains the finiteness/countability property, the proof and ϵ-argument could be carried over to this modified formulation.

C. Discounted MDP

The formulation in Section III-B assumes that the attacker is persistent – he will keep trying to attack the defender's system without giving up. Certainly, this assumption is not realistic in practical scenarios since the attacker might have certain constraints such as time, resources, money, etc. Hence, in each period of time there is some probability that the attacker may give up the attack. We modify the original model given in Section II by adding a new Give-up state G. For the sake of simplicity, we assume the probability that

attacker may give up, $(1-\rho)$, is the same for any connection-detection state.

The state G is also a natural cost-free state since no future attacks will take place once the attacker gives up. In view of an MDP formulation, G could be put into the set of termination states T. It is easy to see that this modified model is equivalent to a *discounted* version of the MDP formulated in Section III-B. Therefore we can express the cost vector $\mathbf{v}_\pi$ as $v_\pi(s) = \sum_{k=0}^{\infty} \rho^k \mathrm{E}_\pi [R_k | S_0 = s]$ where the scalar $\rho \in (0, 1)$ is the *discount factor*. Note here the discount factor could reflect a combination of the chance attacker giving up and the actual discounting of future costs from the defender's perspective. Nevertheless, we only focus on the multiple goals case since there is no natural immediate function in discounting sense for the single goal formulation.

If the discount factor $\rho = 1$ the problem is equivalent to the MDP analyzed in Section III-B. However, the Never-Expel policy is not necessarily optimal for other values of ρ. Expelling the attacker might be beneficial for $0 < \rho < 1$ since the defender is able to avoid immediate costs for some period of time during which the attacker may give up. In other words, "postponing" the attack by expulsion could be a reasonable strategy now. This differs from the undiscounted case where the future costs are considered with their exact amount.

The above discussion leads to two intuitive strategies for the defender in the presence of the discount factor: (1) postponing the attack by expulsion once he detects the existence of the attacker since the attacker may give up in the future with positive probability, (2) out-learning the attacker (learn as fast as possible) to thwart attacks by keeping the attacker in the system. Note that (2) will require the defender to bear certain amount of risk before he acquires enough knowledge to counteract the attack totally. Thus, there will be an interesting trade-off when the defender optimizes his policy. Unfortunately, though numerical study shows that the optimal policy exhibits certain structure, it could not be characterized analytically in general. More discussion on the discounted MDP formulation is provided in Section IV-B.

IV. Numerical Results

This section studies the proposed model numerically. We first present some simulation results of the stochastic model proposed in Section II. Then, some observations about the discounted MDP model is made.

A. Simulation Results

Without formulating the model into an MDP, one can simulate the evolution of the attack-defense process under different defender policies. To begin with, it is interesting to compare two extreme policies Always-Expel and Never-Expel. In Figure 3, 4 the result of a typical sample path is displayed for the geometric case. The performance measure is defined as the cumulative probability of attacker success (the single goal formulation), and the parameter choice is $\alpha = .02, \beta = .05, \gamma = .01, \varepsilon = .05, \delta = .05$. Besides, we assume at the initial state $(k = 0)$ both attacker and defender's have zero knowledge and the attacker is not connected (state (NC, ND)), and simulate the system from time $k = 0$ to 1000. The left plot of Figure 3 shows

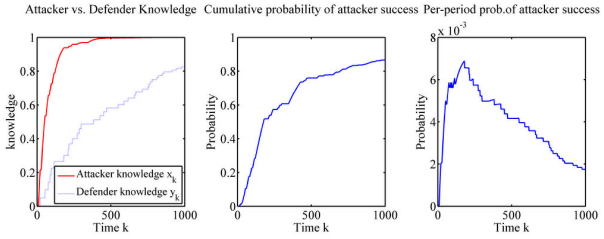

Figure 3: The performance of Always-Expel policy.

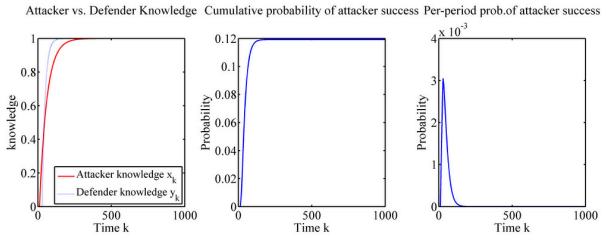

Figure 4: The performance of Never-Expel policy.

that the attacker's knowledge grows faster than that of the defender under the Always-Expel policy, and the middle plot indicates that the cumulative probability of attacker success exceeds 90%. Similar plots are provided in Figure 4 for the Never-Expel policy. By comparing the left plots of Figure 3 and 4, one could see that the defender's knowledge grows faster in the latter and the defender successfully *out-learn* the attacker. Consequently, the cumulative probability of attacker success is under 12%, an order of magnitude of improvement from the rather naive Always-Expel policy. The right plots in both figures demonstrate the evolution of the per-period probability of attacker success. One could observe the rapid defender learning under the Never-Expel policy results in the drastic drop of the per-period attacker success probability which further explains the advantage of the out-learning strategy.

To make the above observations more concrete, we also consider the performance of difference policies on average. We summarize the cumulative probabilities of attacker success of 500 simulation runs under three different policies: Always-Expel, Never-Expel and Random-Expel. The Random-Expel policy is defined as the defender chooses to expel the attacker or not based on the result of tossing a fair coin. Of course, this Random-Expel policy does not have much practical value. However, as one could see in Figure 5 the Always-Expel policy is even beaten by this rather frivolous strategy on average. Other parameters remain the same as above.

B. Discounted MDP

Using the well-known algorithm of *value/policy iteration* [12], we study the optimal policy of the discounted MDP introduced in Section III-C numerically in this section.

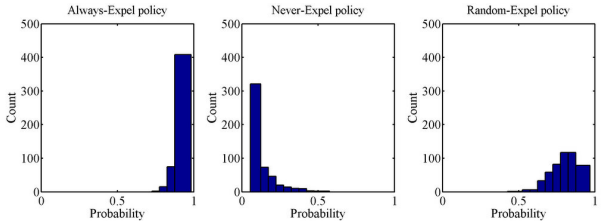

Figure 5: The average performance of 3 different policies: Always-Expel, Never-Expel and Random-Expel. For each policy, we plot the histogram of cumulative probabilities of attacker success over 500 sample paths.

1) Structure of the Optimal Policy: By way of policy iteration, it is easy to compute the optimal stationary policy consisting of decisions only depends on states. Figure 6 illustrates the optimal policy of the discounted MDP under the parameter choice: $\rho = .89, \alpha = .09, \beta = .13, \gamma = .05, \varepsilon = .05, \delta = .05$. The optimal policy is displayed in a control-matrix form where each entry corresponds to the optimal decision of the defender in state $(x, y, \mathbf{3})$.

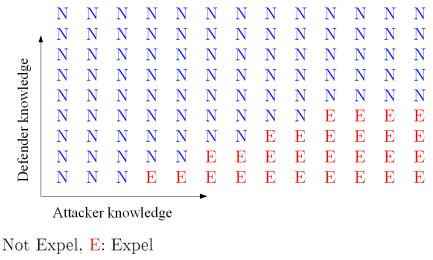

Figure 6: The optimal policy in control-matrix form.

By experiments with various parameter settings, we observe that the optimal policy of the discounted MDP always possesses a lower-triangular, threshold-like structure. Roughly, with a fixed amount of knowledge the defender shall switch from Not-Expel to Expel as the attacker knowledge grows. Note this is similar to the idea of a *threshold policy* where the optimal control switch from one to another as the state exceeds some threshold point. This is quite intuitive in that for fixed defender knowledge, the more the attacker knows about the defender's system, the more immediate damage he may impose. Consequently, it might be more preferable to expel the attacker and avoid relatively significant immediate costs for a while; moreover, since there is some positive probability that the attacker may give up in each period, postponing the attack by expulsion is more advantageous than the out-learning strategy. On the other hand, as the defender knowledge grows larger there are less Expel entries in the optimal control-matrix. Again, this could be understood from the fact that the immediate cost is decreasing w.r.t the defender knowledge.

2) Parameter Effects: We also studied how the threshold ("switching boundary") in the optimal control-matrix varies with respect to different parameters. Under the linear case with fixed learning parameters α, β, we use policy iteration to calculate the optimal control matrix and study the effects of three parameters: the discount factor ρ, the connecting probability ε and the detecting probability δ. In Figure 7, we plot the regions in the optimal control-matrix where the defender shall adopt Expel controls (referred as "expulsion region"). In each subplot, a set of expulsion regions are plotted by varying one parameter and fixing the other two.

From the left subplot in Figure 7, we could see that as the discount factor ρ increases (i.e. the give-up probability decreases), the expulsion region shrinks and the defender shall choose to retain the attacker under most circumstances. As the persistence of the attacker ρ increases, he is more likely to re-connect and attack again. Therefore, postponing the attack is less beneficial and the out-learning strategy becomes more advisable.

In view of the connecting probability ε, as the defender has a stricter access control mechanism (ε increases), the expected time that takes the attacker to get back into the system will be longer, thereby increasing the chance of attacker's withdrawal. Hence, postponing the attack by expulsion is preferable. This effect is demonstrated in the middle subplot of Figure 7.

The detecting probability δ is jointly decided by the defender's intrusion detection system (IDS) and the attacker's technical skill. If the IDS is relatively weak, once connected to the system, the attacker may evade detection for a long period of time and cause considerable costs to the defender without being noticed. Consequently, it is not wise for the defender to choose expulsion under most circumstances if there is not an effective IDS in place. The right subplot in Figure 7 supports this argument, and the reader could observe that as the detecting probability δ drops the expulsion region becomes smaller in the optimal control-matrix.

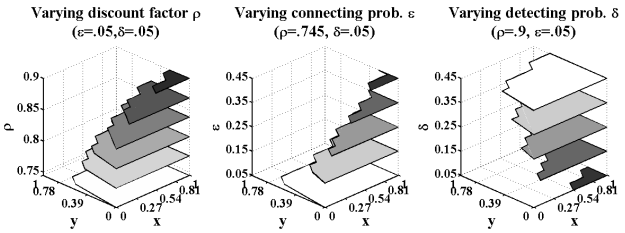

Figure 7: Expulsion region in optimal control-matrix under various parameters values. In each subplot, the expulsion regions and switching boundaries are illustrated by varying one parameter and fixing the other two. ($\alpha = .09, \beta = .13, \gamma = .05$)

C. Multiple Defenders Extension

We also extend our model to a two-defender case. Consider a enterprise's information system consisting of two identical subsystems with exactly the same access control mechanism and IDS. Each subsystem is supervised by one administrator. The learning process and the cost formulations are defined as aforementioned. In addition, we assume that the attacker who tries to infiltrate the system will pick the "weaker" administrator (defender) with less knowledge. We use $y^i, i = 1, 2$ to denote defender i's knowledge. The connection-detection dynamics now have 5 states

$\{(NC, ND), (C^1, ND), (C^1, D^1), (C^2, ND), (C^2, D^2)\}$
where the superscripts denote the subsystem.

If the attacker is persistent, as discussed before, the defender under-attack will find it optimal to retain the attacker in his subsystem. Thus, the problem reduces to a single-defender case. Now we consider an impersistent attacker and consider two cases: (i) the defenders completely share what they know about the attacker so essentially they become a single defender; (ii) they only share how much they know (knowledge level). Considering case (ii), we see that the optimal policy will differ from that of the single-defender case by the following reasoning. Suppose defender 1 is under-attack and has more knowledge than the other defender $y^1 > y^2$. Compared to situation in which defender 1 was the only defender, having a second weaker defender makes it less attractive for defender 1 to expel the attacker. This is because once the attacker is expelled, he will attack the less-knowledgeable defender 2, whereas in the single-defender model the attacker would re-attack defender 1. Consequently the size of the expulsion region (for each y^2, the "expulsion region" is the set of (x, y^1) for which expelling is optimal) gets smaller with smaller y^2. Figure 8 demonstrates this by showing the optimal policy, found by policy iteration, for parameters $\alpha = .09$, $\beta = .13$, $\gamma = .05$, $\varepsilon = .05$, and $\delta = .05$.

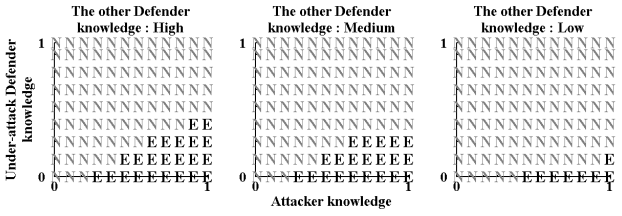

Figure 8: Shrinkage of the expulsion region in the under-attack defender's system-wide optimal control-matrix.

V. Conclusion and Future Work

We have considered the defender's policy optimization problem with presence of the learning effect in a security context. By formulating the problem into a Markov decision process, we are able to analyze the characteristics of the optimal policy. If the attacker's is persistent, the optimal strategy of the defender is to keep the attacker in the system in order to out-learn him and eventually thwart the attacks, which is quite different from the conventional idea of expelling the attacker whenever detecting his presence. For the case where the attacker may give up, it can be formulated into a discounted MDP, and we observe that the optimal policy in this case has certain structure by numerical experiments.

Our model, although quite stylized, is able to capture the interesting effects when one considers the learning effect in a cyber-defense scenario. It demonstrates the potential benefit of gathering intelligence from the attacker during the course of a defense. This idea yields a new perspective in studying the network security problems.

In the future, we plan to extend our model by introducing attacker strategies such as actively disconnecting and re-entering the system in order to avoid the defender's intrusion detection mechanisms. By the techniques from Stochastic Game theory, we hope to investigate the strategies of both the defender and attacker's, which might bring more insights. Besides, knowledge is abstracted as a one-dimensional variable in the current model for the sake of tractability. However, it might be more reasonable to develop a model where knowledge is represented as a multi-dimensional vector indicating various aspects of the knowledge about one's opponent. Instead of using simple learning curves, we also plan to improve the modelling of the learning process (for example, using Bayesian updating) to better capture the nature of learning in the cyber-defense context. This is another direction we plan to pursue in future research.

Appendix

A. Proof of Proposition 1

Proof: As mentioned in Section III-B, the *Bellman's equation* holds for the optimal cost vector $\mathbf{v}^*$. We could write it in terms of the three set of states in our formulation as follows:

$$\begin{aligned}
v^*(x, y, \mathbf{1}) &= 0 + \varepsilon v^*(x, y, \mathbf{2}) + (1 - \varepsilon) v^*(x, y, \mathbf{1}),\\
v^*(x, y, \mathbf{2}) &= r(x, y) + \delta v^*(x + \alpha, y, \mathbf{3})\\
&\quad + (1 - \delta) v^*(x + \alpha, y, \mathbf{2}),\\
v^*(x, y, \mathbf{3}) &= r(x, y) + \min_a \big\{ v^*(x + \alpha, y + \beta, \mathbf{1}),\\
&\quad v^*(x + \alpha, y + \beta, \mathbf{3}) \big\},
\end{aligned}$$

for all $x \in X, y \in Y$, where $a \in \{\text{Expel}, \text{Not-Expel}\}$ denotes the defender's action. Besides, note that $v^*(x, y, \mathbf{1}) = v^*(x, y, \mathbf{2}), \forall x, y$ from the first expression. Consequently, we will use $v^*(x, y, \mathbf{2})$ in place of $v^*(x, y, \mathbf{1})$ in the sequel.

Recall that state space consists of three grids indexed by the set $\{(NC, ND), (C, ND), (C, D)\}$ and each grid is formed by the Cartesian product of two sets $X = \{0, \alpha, 2\alpha, \ldots, 1\}, Y = \{0, \beta, 2\beta, \ldots, 1\}$ (Cf. Figure 2). Let $\tilde{y} := \max\{y : y < 1, y \in Y\}$. Moreover, the set of termination states T are cost-free therefore $v^*(x, 1, \cdot) = 0$. Consider the equation $v^*(x, \tilde{y}, \mathbf{3}) = r(x, \tilde{y})$ where we use the fact $v^*(x, 1, \mathbf{3}) = 0$ to eliminate the minimization operator. Obviously, since we have $\partial r(x, y)/\partial x > 0$ it follows that $v^*(x, \tilde{y}, \mathbf{3})$ is monotonically increasing w.r.t x for $\tilde{y}$.

Next, consider the optimal costs $v^*(x, y, \mathbf{2})$. We obtain the following expression by expanding $v^*(\cdot, y, \mathbf{2})$ terms iteratively:

$$\begin{aligned}
v^*(x, y, \mathbf{2}) &= \sum_{i=0}^{\infty} (1 - \delta)^i r(x + i\alpha, y)\\
&\quad + \delta \sum_{i=0}^{\infty} (1 - \delta)^i v^*(x + (i + 1)\alpha, y, \mathbf{3}), \quad (1)
\end{aligned}$$

If we apply this expression to $v^*(x, \tilde{y}, \mathbf{2})$ and $v^*(x', \tilde{y}, \mathbf{2})$ for $x > x'$, it then follows that $v^*(x, \tilde{y}, \mathbf{2}) > v^*(x', \tilde{y}, \mathbf{2})$ since in view of (1) they are both monotonically increasing w.r.t x. This is true since $\partial r/\partial x > 0$ and we have shown that $\partial v^*(x, \tilde{y}, \mathbf{3})/\partial x > 0$. Nevertheless, one could index the

elements in sets X, Y and replace increment expressions such as $(x + i\alpha)$ with corresponding indexed elements in X, Y in the above derivation. Then the proof given here could be applied to the slightly more general extensions as described in Section III-B.

So far we have established the properties (a) and (b) in Proposition 1 for $\tilde{y}$. Consider the second summation in (1) and observe that $\delta \sum_{i=0}^{\infty}(1-\delta)^i v^*(x+(i+1)\alpha, \tilde{y}, \mathbf{3}) > \delta \sum_{i=0}^{\infty}(1-\delta)^i v^*(x, \tilde{y}, \mathbf{3}) = v^*(x, \tilde{y}, \mathbf{3})$. Consequently, it follows that $v^*(x, \tilde{y}, \mathbf{2}) > v^*(x, \tilde{y}, \mathbf{3})$ which is just the property (c) desired.

Having established property (a),(b) and (c) for $\tilde{y}$ as our base case, we could carry out the inductive step to prove the general case. Assume that the assertion holds for some $y < \tilde{y}$, and consider the case of $(y - \beta)$ for any $x > x'$. For property (a), since $v^*(x, y-\beta, \mathbf{3}) = r(x, y-\beta) + \min_a\{v^*(x, y, \mathbf{2}), v^*(x, y, \mathbf{3})\} = r(x, y-\beta) + v^*(x, y, \mathbf{3})$ by induction hypothesis, clearly we have $v^*(x, y-\beta, \mathbf{3}) > v^*(x', y-\beta, \mathbf{3})$. For property (b), similar to the base case of $\tilde{y}$ demonstrated above, we could have $v^*(x, y-\beta, \mathbf{2}) > v^*(x', y-\beta, \mathbf{2})$ by the monotonicity of $v^*(x, y-\beta, \mathbf{3})$. Finally property (c) follows in view of expression (1) and property (a) for $(y - \beta)$.

Thus, we conclude that Proposition 1 is valid. ■

B. ϵ-argument for the Geometric Case

First we note that the state space for the geometric case is countable. Besides, the immediate cost function $r(x, y) = \gamma x(1-y)$ or $r(x, y) = -\log(1-\gamma x(1-y))$ are non-negative/non-positive respectively. This choice of the immediate cost function satisfies the *Negativity/Positivity* assumption, thus the results of infinite horizon total expected cost problem could be applied directly to our problem (Cf. [12], vol. 2, chap. 3). Most importantly, the *Dynamic Programming mapping* has a fixed point as the cost vector $\mathbf{v}_\pi$:

$$
\begin{aligned}
v_\pi(x_m, y_n, \mathbf{1}) &= \varepsilon v_\pi(x_m, y_n, \mathbf{2}) + (1-\varepsilon) v_\pi(x_m, y_n, \mathbf{2}),\\
v_\pi(x_m, y_n, \mathbf{2}) &= r(x_m, y_n) + \delta v_\pi(x_{m+1}, y_n, \mathbf{3})\\
&\quad + (1-\delta) v_\pi(x_{m+1}, y_n, \mathbf{2}),\\
v_\pi(x_m, y_n, \mathbf{3}) &= r(x_m, y_n)\\
&\quad + \begin{cases} v_\pi(x_{m+1}, y_{n+1}, \mathbf{1}) & \text{if Expel}\\ v_\pi(x_{m+1}, y_{n+1}, \mathbf{3}) & \text{if Not-Expel} \end{cases},
\end{aligned}
$$

where $x_m = 1-(1-\alpha)^m, y_n = 1-(1-\beta)^n$. The subscripts m, n could be viewed as the indices of elements in set X, Y.

Note that $v_\pi(x_m, y_n, \cdot)$ are linear functions of r, and we can express $v_\pi(x_m, y_n, \cdot)$ as a linear summation in terms of r : $v_\pi(x_m, y_n, \cdot) = \sum_{i=m}^{\infty}\sum_{j=n}^{\infty}\lambda_{i,j} r(x_i, y_j)$ where $0 < \lambda_{i,j} < 1$ are the proper coefficients [1]. One could observe the following:

$$
\begin{aligned}
v_\pi(x_m, y_n, \cdot) &\le \sum_{j=n}^{\infty}\lambda_j r(1, y_j) && (2)\\
&\le \Lambda \sum_{j=n}^{\infty}\gamma(1-\beta)^j && (3)\\
&= \frac{\Lambda\gamma}{\beta} - \Lambda\gamma\sum_{j=0}^{n-1}(1-\beta)^j. && (4)
\end{aligned}
$$

We use the fact that $r(x, y) \le r(1, y)$ and define $\lambda_j := \sum_{i=m}^{\infty}\lambda_{i,j}$ to obtain (2); and in (3) we let $\Lambda := \max_j \lambda_j$ and plug in $r(x, y) = \gamma x(1-y)$. Equality (4) is arrived by noticing $\sum_{i=n}^{\infty}(1-\beta)^j = \sum_{i=0}^{\infty}(1-\beta)^j - \sum_{i=0}^{n-1}(1-\beta)^j$.Then, it is clear that $v_\pi(x_m, y_n, \cdot)$ is upper-bounded by (4) which is decreasing w.r.t n. Therefore, we could always find a large enough N to make $v_\pi(x_m, y_n, \cdot) < \epsilon$ for arbitrary $\epsilon > 0$.

Furthermore this ϵ-argument is also applicable for the multiple goals formulation $r(x, y) = -\log(1-\gamma x(1-y))$. Consider the relation:

$$
\Lambda\sum_{j=n}^{\infty}(-\log(1-\gamma(1-\beta)^j)) \le \Lambda\int_n^{\infty}(-\log(1-\gamma(1-\beta)^j),
$$

where the function $-\log(1-\gamma(1-\beta)^j))$ decreases geometrically fast towards 0. Again, since the r.h.s of the preceding inequality is bounded, $v_\pi(x, y_n, \cdot)$ is upper-bounded as well.

[1] Note that $\lambda_{i,j}$ is determined by the given starting state $s = (x_m, y_n, \cdot)$. Refer to expression (1) for a similar expansion to see that the coefficients are indeed bounded as well.

REFERENCES

[1] H. Project, *Know Your Enemy: Learning About Security Threats*, 2nd ed. Addison Wesley, 2004.

[2] M. Dacier and Y. Deswarte, "Privilege graph: an extension to the typed access matrix model," in *ESORICS '94: Proceedings of the Third European Symposium on Research in Computer Security*. London, UK: Springer-Verlag, 1994, pp. 319–334.

[3] M. Dacier, Y. Deswarte, and M. Kaniche, "Quantitative assessment of operational security: Models and tools," LAAS Research Report 96493, 1996.

[4] R. Ortalo, Y. Deswarte, and M. Kaaniche, "Experimenting with quantitative evaluation tools for monitoring operational security," *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 633–650, 1999.

[5] S. Jha, O. Sheyner, and J. Wing, "Two formal analysis of attack graphs," *Computer Security Foundations Workshop, IEEE*, vol. 0, p. 49, 2002.

[6] K. wei Lye and J. M. Wing, "Game strategies in network security," *International Journal of Information Security*, vol. 4, pp. 71–86, 2005.

[7] K. Sallhammar, "Stochastic models for combined security and dependability evaluation," Ph.D. dissertation, Norwegian University of Science and Technology, Faculty of Information Technology, Mathematics and Electrical Engineering, 2007.

[8] T. Alpcan and T. Başar, "A game theoretic approach to decision and analysis in network intrusion detection," in *2003. Proceedings. 42nd IEEE Conference on Decision and Control*, vol. 3, 2003, pp. 2595 – 2600.

[9] ——, "A game theoretic analysis of intrusion detection in access control systems," in *2004. Proceedings. 43nd IEEE Conference on Decision and Control*, vol. 2, 2004, pp. 1568– 1573.

[10] ——, "An intrusion detection game with limited observations," in *Proccedings of 12th International Symposium on Dynamic Games and Applications*, 2006.

[11] J. Filar and K. Vrieze, *Competitive Markov Decision Processes*. Springer, 1996.

[12] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed. Athena Scientific, 2001, vol. I, II.