INFORMS Journal on Computing
Vol. 20, No. 1, Winter 2008, pp. 112–123
ISSN 1091-9856 | EISSN 1526-5528 | 08 | 2001 | 0112

DOI 10.1287/ijoc.1070.0222

Intrusion-Detection Policies for IT Security Breaches

Hulisi Ogut
Department of Business Administration, TOBB University of Economics and Technology,
Söğütözü, Ankara 06560, Turkey, hogut@etu.edu.tr

Huseyin Cavusoglu, Srinivasan Raghunathan
The School of Management, University of Texas at Dallas, Richardson, Texas 75083
{huseyin@utdallas.edu, sraghu@utdallas.edu}

Intrusion-detection systems (IDSs) form an important component of IT security architectures, but the low proportion of hackers in the user population severely limits the usefulness of IDSs. Thus, even when the IDS is good, an intrusion signal may not imply that the user is more likely to be a hacker than a normal user. Ignoring the low base rate for the proportion of hackers results in acting on every intrusion signal, which is costly because of the high rate of false alarms. This problem is known as the *base-rate fallacy* in IDSs. On the other hand, ignoring intrusion signals renders IDSs useless. We propose and analyze waiting-time policies, which specify a response to signals from IDSs. We formulate the problem as a stochastic dynamic programming model and derive the optimal waiting time before acting upon an intrusion signal. Because the optimal policy is difficult to implement in many situations, we also derive and theoretically analyze a myopic policy. Our simulations suggest that the behavior of the myopic policy is qualitatively similar to that of the optimal policy. Further, the myopic policy performs better than other policies often used in practice, such as the Bayes policy and m-strike policies. The myopic policy can be implemented easily in a decision support system that supplements an IDS to mitigate the base-rate fallacy and to improve the value of the IDS.

Key words: IT security; dynamic programming; stochastic model applications; decision analysis
History: Accepted by Ramayya Krishnan, former Area Editor for Telecommunications and Electronic Commerce; received June 2004; revised August 2005, January 2006, May 2006, November 2006, January 2007; accepted January 2007. Published online in *Articles in Advance* December 3, 2007.

1. Introduction

IT-based controls are deployed to secure information assets. Intrusion-detection systems (IDSs) are one such control, which complement preventive controls such as firewalls and anti-virus software, by detecting intrusions in real time. While preventive controls aim to stop intrusions, IDSs attempt to detect intrusions by external hackers that have passed through preventive controls and intrusions by insiders, who often form a major part of the hacker population (Escamilla 1998, Russell and Gangemi 1992).

An IDS detects intrusions by analyzing user log files and network streams in user traffic. It generates an intrusion signal when it detects an event that matches a predefined pattern associated with known intrusions (signature-based detection) or an event that varies significantly from the normal profile of a user or a system (anomaly-based detection). The effectiveness of an IDS is measured using two parameters: the likelihood of getting an intrusion signal when an intrusion occurs (true positive or true detection rate) and the likelihood of getting an intrusion signal when there is no intrusion (false positive or false alarm rate). The effectiveness depends on various design factors, such as the technology used (signature-based versus anomaly-based), design specifications (e.g., the acceptable noise level in an anomaly-based system), and the configuration (strict versus loose).

Security experts have questioned the usefulness of IDSs in protecting information assets. Gartner (2003) dismisses IDSs as failed technology and recommends allocation of all security budgets to preventive controls, claiming that IDSs suffer from low detection and high false alarm rates. Because hackers form only a small part of the user population, an IDS with even a low or moderate false alarm rate generates more intrusion signals for normal users than for hackers, thus making IDS signals less useful. Ignoring the prior probability and acting on every IDS intrusion signal, known as the *base-rate fallacy* (Axellson 2000) is costly. On the other hand, ignoring intrusion signals renders the IDS useless. The proponents of IDSs claim that because technology that completely prevents intrusions does not exist, IDSs are the only efficient mechanism to deal with intrusions that have already occurred (Shipley 1999), and thus should be deployed even with their inherent limitation.

The base-rate fallacy stems from Bayes' theorem ($A_1, A_2, \ldots, A_n$ partition the sample space):

$$P(A_i \mid B) = P(A_i)\frac{P(B \mid A_i)}{\sum_{j=1}^{n} P(A_j)P(B \mid A_j)}$$
$$= \text{Prior Probability}$$
$$\times \text{Strength of Signal about } A_i.$$

Consider two types of users: normal and hackers. Assume $P(\text{hacker}) = 1/1{,}000$, a very high estimate in reality, $P(\text{intrusion-signal} \mid \text{hacker}) = P(\text{no-intrusion-signal} \mid \text{normal}) = 0.75$, typical for a high-quality IDS (Axelsson 2000). Thus, $P(\text{hacker} \mid \text{intrusion-signal}) \cong 0.003$. This probability drops significantly when the quality of the IDS degrades, or the proportion of hackers in the population decreases. A low value for $P(\text{hacker} \mid \text{intrusion-signal})$ means that acting on every intrusion signal will be costly, but ignoring intrusion signals from the IDS goes against the very purpose of deploying it. For a perfect IDS—a 100% detection rate and a zero false-positive rate—the base-rate fallacy disappears. In fact, designing a perfect IDS is one of the goals of security researchers. However, limitations of current detection technology make it impossible to design a perfect IDS, so other mechanisms to maximize the value with a less-than-perfect IDS should be deployed.

We propose a solution to mitigate the negative effects arising from the low base rate of the proportion of hackers. Instead of acting on an intrusion signal immediately or ignoring the intrusion signal completely, we may choose to wait to get additional information and make a more informed decision later. Waiting mitigates the base-rate fallacy: The estimate of the probability that the user is a hacker increases (decreases) if the user is indeed a hacker (normal user) when we delay action. While waiting may reduce the chance of taking an incorrect action, it also allows a hacker to cause more damage, so the problem of how long to wait becomes critical. Our policy can be implemented as a decision support system (DSS) that uses the IDS signals as input to make a recommendation about when to take action against a user.

Our policies stipulate action when we get an intrusion or a no-intrusion signal from the IDS. We formulate the problem as a stochastic dynamic programming model and derive the optimal solution. Because the optimal policy may be difficult to implement, we also derive and analyze a myopic policy; the myopic policy waits longer before taking action. The waiting time under the myopic policy is higher (lower) when (i) the cost of taking action against a normal user is higher (lower), (ii) the cost of waiting to take action against a hacker is lower (higher), or (iii) the proportion of hackers in the user population is lower (higher). The myopic policy waits longer when hackers take more frequent actions; however, it waits longer when normal users take more frequent actions only if the waiting time is short. The optimal waiting time increases (decreases) as we become more risk-averse if the ratio of the cost of waiting to the cost of taking action against a normal user is sufficiently large (small). The optimal policy is qualitatively similar to that of the myopic policy, so our analysis of the myopic policy is likely to hold for the optimal policy too.

Our analysis is restricted largely to the myopic policy. A comparison of the myopic policy with other policies such as the simple Bayes policy and the commonly used m-strike policy, which refers to the policy of taking action after getting m intrusion signals from the IDS, proved to be intractable, so we numerically compared the myopic policy with Bayes and m-strike policies. The myopic policy usually performed better than these policies. There is benefit to using the myopic policy when deploying a less-than-perfect IDS, and the myopic policy is effective in mitigating the base-rate fallacy.

In §2, we review the literature, and describe our model in §3. In §4, we derive the optimal policy. In §5, we study the myopic policy and analyze its properties analytically. We compare the myopic policy, Bayes policy, and m-strike policies numerically in §6. We conclude in §7.

2. Literature Review

Since Denning (1987), researchers have investigated IDSs, primarily from a technical perspective in two broad streams: *design* and *configuration*. The first stream has focused on developing and improving algorithms (both signature-based and anomaly-based) to detect intrusions. For signature-based systems, see Porras and Kemmerer (1992), Lunt (1993), and Kumar and Spafford (1996). For anomaly-based systems, see D'haeseleer et al. (1996), Porras and Neumann (1997), Neumann and Porras (1999), and Zamboni and Spafford (1999). Several studies, including the well-known Defense Advanced Research Projects Agency (DARPA) study, evaluated the performance of IDSs (Lippman et al. 2000) using controlled experiments. These studies used detection rate and false alarm rate as performance measures. The main goal of technical research on IDSs has been to increase the detection rate for a given false-positive rate, or reduce the false-positive rate for a given detection rate. Because costs of false-positive and false-negative errors are different, Lee et al. (2001) proposed an adaptive algorithm that incorporates cost elements associated with giving an intrusion signal.

The second stream is more recent. Because there may be different tolerance levels for false positives

and intrusion detection, researchers have begun to investigate how to configure a given IDS accordingly (Cavusoglu et al. 2005). Ulvila and Gaffney (2004) proposed a decision analysis approach, and Cavusoglu and Raghunathan (2004) compared decision analysis and game-theoretic approaches. We assume that an IDS with a specific configuration is given and address how to use the given IDS, i.e., when to take an action following a signal from the IDS, to minimize the cost of dealing with intrusions.

In medical testing, Ozekici and Pliska (1991) analyzed scheduling diagnostic inspections of a patient whose health can deteriorate according to a Markov process. In reliability, Diamond (1982) considers an inspector, with a finite number of inspections at his disposal, who aims to detect fraud by the inspectee. The loss incurred is a function of the undetected time period, as in our model.

3. Model Description

An IDS analyzes each user action and generates a signal. A user action involves a system-level action, such as invoking a system command, which is recorded in the user log file. The prior probability that a user is a hacker is p_0. User actions arrive in independent Poisson processes with rate λ_H for hackers and λ_N for normal users (Jonsson and Olovsson 1997, Moitra and Konda 2000). Typically $\lambda_H \leq \lambda_N$ because hackers tend to analyze carefully results of their past actions before deciding their future course of action. All hacker actions are intrusive, and all actions of a normal user are benign. After every user action, the IDS signals whether the user's action is an intrusion. We use $S_i = 0$ and $S_i = 1$, respectively, to denote the intrusion signal and the no-intrusion signal after the ith user action. The quality of the IDS is determined by its classification accuracy. Let $p(S_i = 1 \mid \text{Hacker}) = \theta_i^H$ and $p(S_i = 0 \mid \text{Normal User}) = \theta_i^N$. The signals from the IDS for a particular user are independent; i.e., the value of a signal after a user action depends only on whether the user is a hacker or a normal user and not on prior signals generated for the same user. Dependence between signals over time will change only the probability-updating process, captured by (1) and (2) below in §4. We assume that the IDS performs at least as well as random guessing, i.e., $\theta_i^H, \theta_i^N \geq 1/2$. Let p_i be the firm's posterior probability that the user is a hacker given the signal history up to the ith signal from the IDS, which occurs at time t_i.

In response to a signal from the IDS, we may take an action (e.g., terminating the user's session) against the user. A cost c_N is incurred when an action is incorrectly taken against a normal user. Undetected hacking activity incurs a cost c_H that is a nondecreasing convex function of time, i.e., $\partial c_H/\partial t, \partial^2 c_H/\partial t^2 \geq 0$. The functional forms we use for the damage caused by a hacker and for the IDS quality in detecting intrusions assume that we restrict our analysis to only one type of intrusion. Another way of interpreting this assumption is that all types of intrusions are equally likely to be detected and cause the same amount of damage. It may be impossible to assess or to recognize the actual damage, but we can estimate it.

The goal is to determine the optimal time to wait before concluding that the user is a hacker and terminating the user's session. We use the following waiting-time strategy. After each signal from the IDS, we update the posterior probability that the user is a hacker (or a normal user) and compute its optimal waiting time. Let f_i be the waiting time (grace period) after the ith user action. If the user takes no new action (so the IDS gives no more signals before f_i elapses), we conclude that the user is a hacker and terminate the session. If the IDS gives a signal before the grace period elapses, we update the posterior probability and compute f_{i+1}. For example, in Figure 1, we take no action after observing the ith signal (intrusion signal or no-intrusion signal), which arrives at time t_i; instead, we set a grace period of f_i. Because this time does not elapse by the time the next user action is observed at t_{i+1}, we set a new grace period f_{i+1}. The action is taken at $t_{i+1} + f_{i+1}$ because signal $i+2$ does not arrive before $t_{i+1} + f_{i+1}$. The process is repeated until we either conclude that the user is a hacker and terminate the session, or the user leaves the system.

We derive the optimal waiting time by maximizing the expected future utility (or minimizing the expected future disutility). The disutility, as a function of cost, depends on the risk profile, for which we use the power function $V(c) = c^\beta$, where c is the expected cost, and $\beta > 0$ is the risk parameter. $\beta > 1$, $\beta = 1$, and $\beta < 1$, respectively, capture risk aversion, risk neutrality, and risk seeking. In the IT security context, we would expect either risk aversion or risk neutrality, so assume $\beta \geq 1$. Table 1 gives some additional notation.

We use a decision-theoretic model to determine the waiting-time policy; i.e., we assume that users do not change their actions in response to the policy.

4. Optimal Policy

In this section, we derive the optimal waiting policy, f_i^*, by minimizing the expected disutility during the

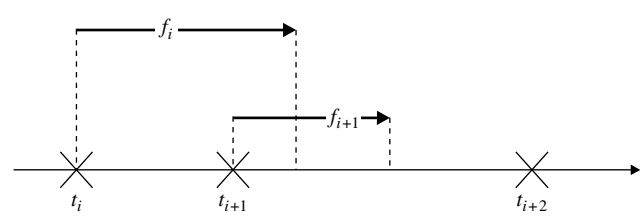

Figure 1 **Illustration of Detection Procedure**

Table 1 Additional Notation

$\hat{S}_i$	Vector of signals until and including the ith signal
$C_H(\hat{S}_i, t_i)$	Expected cost during (t_i, ∞) if the user is a hacker
$C_N(\hat{S}_i, t_i)$	Expected cost during (t_i, ∞) if the user is a normal user
$V(\hat{S}_i, t_i)$	Expected disutility during (t_i, ∞). Later, we use V with subscripts N and H to indicate the expected utility for a normal user and a hacker, respectively.

period $[t_i, \infty)$. If we wait for f_i units of time when we receive the ith signal, then two cases can occur: (i) A new signal does not arrive before $t_i + f_i$, resulting in the conclusion at time $t_i + f_i$ that the user is a hacker, or (ii) a new signal arrives before $t_i + f_i$, causing computation of a new optimal waiting policy, f^*_{i+1}, based on the new information. The decision tree for computing the expected disutility during $[t_i, \infty)$ is in Figure 2. The expression at a leaf node of the tree represents the firm's disutility at that node.

In scenario (i), the user is a hacker, and we correctly identify that and take action at time $t_i + f_i$. Thus, the cost incurred during $[t_i, \infty)$ is $c_H(t_i + f_i) - c_H(t_i)$, which translates into a disutility of $(c_H(t_i + f_i) - c_H(t_i))^\beta$. In scenario (ii), the user is a hacker, and a new no-intrusion signal is generated by the IDS before the end of the grace period. In this case, we compute f^*_{i+1} at t_{i+1}. The expected disutility during $[t_i, \infty)$ is $(c_H(t_{i+1}) - c_H(t_i) + C_H(\hat{S}_i \cup \{0\}, t_{i+1}))^\beta$. We incur cost $c_H(t_{i+1}) - c_H(t_i)$ during $[t_i, t_{i+1})$, and the last term is the cost incurred during $[t_{i+1}, \infty)$. In scenario (iii), the user is a hacker and a new intrusion signal is generated by the IDS before we take action. Similar to scenario (ii), we compute f^*_{i+1} at t_{i+1} in this scenario. The disutility during $[t_i, \infty)$ is $(c_H(t_{i+1}) - c_H(t_i) + C_H(\hat{S}_i \cup \{1\}, t_{i+1}))^\beta$. Scenarios (iv)–(vi) capture similar possibilities for a normal user.

We compute the posterior probability p_i, given the signal history $\hat{S}_i$, using Bayes' rule as

$$p_i = P(\text{Hacker} \mid \hat{S}_i) = p_{i-1} \frac{P(\hat{S}_i \mid \text{Hacker})}{P(\hat{S}_i \mid \text{Normal User})(1 - p_{i-1}) + P(\hat{S}_i \mid \text{Hacker}) p_{i-1}}, \tag{1}$$

which can be written as

$$p_i = \begin{cases} p_{i-1} \dfrac{\theta_i^H}{(1 - \theta_i^N)(1 - p_{i-1}) + \theta_i^H p_{i-1}} & \text{if } S_i = 1 \\ p_{i-1} \dfrac{1 - \theta_i^H}{\theta_i^N (1 - p_{i-1}) + (1 - \theta_i^H) p_{i-1}} & \text{if } S_i = 0. \end{cases} \tag{2}$$

Next, we present the result regarding how the optimal waiting policy affects the base-rate fallacy.

Proposition 1. *p_i is monotonically increasing (decreasing) in i for a hacker (normal user).*

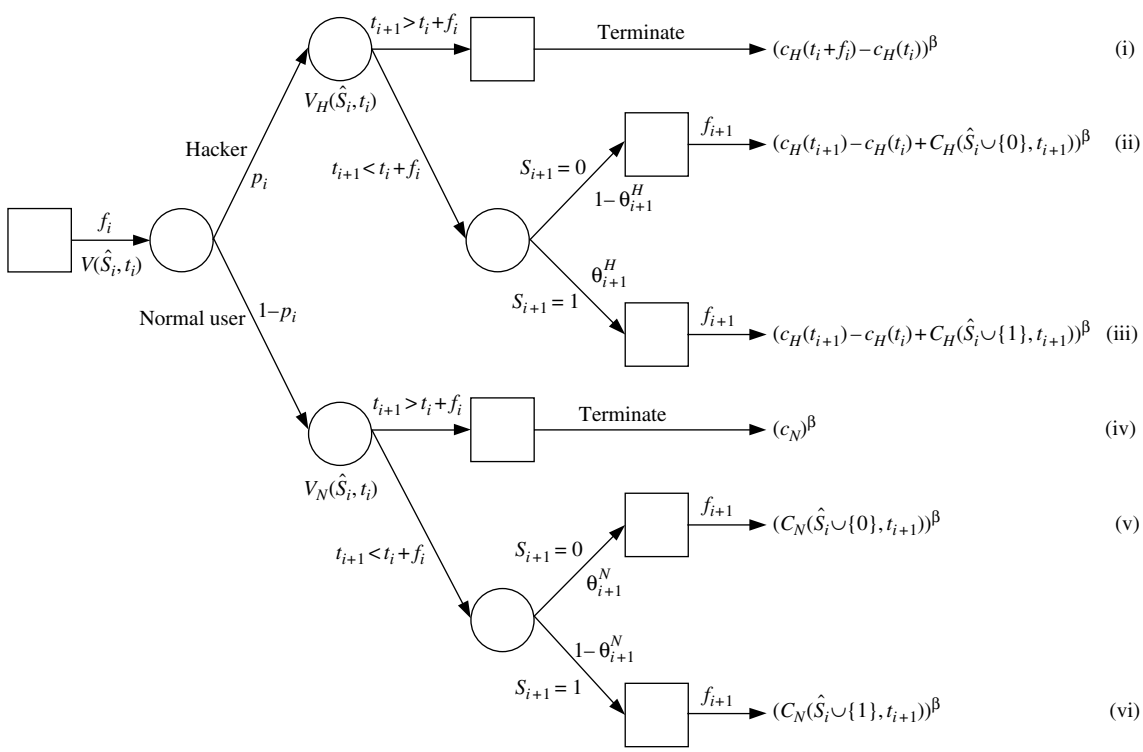

Figure 2 Decision-Tree Representation of Intrusion-Detection Problem at Time t_i

The proofs for all propositions are in the Online Supplement to this paper (available at http://joc.pubs.informs.org/ecompanion.html).

Proposition 1 offers a rationale to wait for additional signals before acting. Waiting improves the expected accuracy of the classification of the user. However, waiting cost is high if the user happens to be a hacker. Consequently, we cannot wait indefinitely and should determine the optimal waiting time.

Using the decision tree, we can write the minimum expected disutility at t_i with a signal history of $\hat{S}_i$ as

$$\begin{aligned}
&V(\hat{S}_i, t_i)\\
&= \min_{f_i} p_i \Bigg\{ e^{-\lambda_H f_i}[c_H(t_i+f_i) - c_H(t_i)]^\beta\\
&\qquad + \int_{t_i}^{t_i+f_i} \lambda_H e^{-\lambda_H(t_{i+1}-t_i)}(1-\theta_{i+1}^H)\\
&\qquad\quad \cdot ([c_H(t_{i+1}) - c_H(t_i)] + C_H(\hat{S}_i \cup \{0\}, t_{i+1}))^\beta \, dt_{i+1}\\
&\qquad \cdot \int_{t_i}^{t_i+f_i} \lambda_H e^{-\lambda_H(t_{i+1}-t_i)} \theta_{i+1}^H([c_H(t_{i+1}) - c_H(t_i)]\\
&\qquad\qquad + C_H(\hat{S}_i \cup \{1\}, t_{i+1}))^\beta \, dt_{i+1} \Bigg\}\\
&\quad + (1-p_i) \Bigg\{ e^{-\lambda_N f_i (c_N)^\beta} + \int_{t_i}^{t_i+f_i} \lambda_N e^{-\lambda_N(t_{i+1}-t_i)}\\
&\qquad\qquad \cdot \theta_{i+1}^N (C_N(\hat{S}_i \cup \{0\}, t_{i+1}))^\beta \, dt_{i+1}\\
&\qquad \cdot \int_{t_i}^{t_i+f_i} \lambda_N e^{-\lambda_N(t_{i+1}-t_i)}(1-\theta_{i+1}^N)\\
&\qquad\qquad \cdot (C_N(\hat{S}_i \cup \{1\}, t_{i+1}))^\beta \, dt_{i+1} \Bigg\}. \qquad (3)
\end{aligned}$$

The model in (3) is a stochastic dynamic program, which may be solved using a backward induction procedure if the parameter values are known. However, as in many stochastic dynamic programs, obtaining an analytical solution may be impossible except for simple functional forms for c_H (Ross 1983). We use a stepwise cost function to illustrate the backward induction procedure next.

In the stepwise function, the cost from a hacking activity increases only when the hacker acts. We chose the stepwise function to illustrate the solution procedure for three reasons. First, it is a reasonable approximation for the cost caused by hacker actions. This approximation states that once a hacker command is executed, the entire damage from that action is incurred instantaneously, and no further damage occurs until the next hacker command. Second, it allows derivation of the analytical solution. Third, any continuous function can be approximated to any degree of accuracy by a well-chosen stepwise function.

For our illustration, we further assume that the hacker takes a maximum of only two actions during a session and leaves the system after the second action. This assumption is not required, but we made it for exposition. The disutility-minimization model at t_0 for the stepwise function is

$$\begin{aligned}
V(\hat{S}_0, t_0) = \min_{f_0} p_0 \int_{t_0}^{t_0+f_0} &\lambda_H e^{-\lambda_H(t_1-t_0)}\\
&\cdot [(1-\theta_1^H)([c_H(t_1) - c_H(t_0)] + C_H(\hat{S}_0 \cup \{0\}, t_1))^\beta\\
&\quad + \theta_1^H([c_H(t_1) - c_H(t_0)] + C_H(\hat{S}_0 \cup \{1\}, t_1))^\beta] \, dt_1\\
&+ (1-p_0)\Bigg(e^{-\lambda_N f_0}(c_N)^\beta + \int_{t_0}^{t_0+f_0} \lambda_N e^{-\lambda_N(t_1-t_0)}\\
&\qquad \cdot [\theta_1^N (C_N(\hat{S}_0 \cup \{0\}, t_1))^\beta + (1-\theta_1^N)\\
&\qquad \cdot (C_N(\hat{S}_0 \cup \{1\}, t_1))^\beta] \, dt_1 \Bigg). \qquad (4)
\end{aligned}$$

The above model uses the fact that, in the stepwise function, $c_H(t_i+f_i) = c_H(t_i)$ when $t_i + f_i < t_{i+1}$. Let s assume a value of 0 or 1 depending on whether IDS generates a nonintrusion or intrusion signal, respectively. The following equalities hold given that the user takes a maximum of only two actions:

$$\begin{aligned}
V_N(\hat{S}_0 \cup \{s\}, t_1) &= e^{-\lambda_N(f_1 \mid S_1 = s)}(c_N)^\beta;\\
V_H(\hat{S}_0 \cup \{s\}, t_1) &= (1 - e^{-\lambda_H(f_1 \mid S_1 = s)})(c_H(t_2) - c_H(t_1))^\beta\\
C_N(\hat{S}_0 \cup \{s\}, t_1) &= e^{-\lambda_N(f_1 \mid S_1 = s)} c_N;\\
C_H(\hat{S}_0 \cup \{s\}, t_1) &= (1 - e^{-\lambda_H(f_1 \mid S_1 = s)})(c_H(t_2) - c_H(t_1)).
\end{aligned}$$

To drive the optimal policy at t_0, we start from the optimal policy at t_1. We will minimize the following expression at t_1:

$$\begin{aligned}
\min_{(f_1 \mid S_1 = s)} &(p_1 \mid S_1 = s) V_H(\hat{S}_0 \cup \{s\}, t_1)\\
&+ (1 - (p_1 \mid S_1 = s)) V_N(\hat{S}_0 \cup \{s\}, t_1).
\end{aligned}$$

Solving the above model, we get the following optimal waiting policy at time t_1.

If $\lambda_N > \lambda_H$,

$$\begin{aligned}
&(f_1^* \mid S_1 = s)\\
&\quad = (\ln(\lambda_N[1 - (p_1 \mid S_1 = s)](c_N)^\beta) - \ln(\lambda_H(p_1 \mid S_1 = s)\\
&\qquad \cdot [c_H(t_2) - c_H(t_1)]^\beta)) \cdot (\lambda_N - \lambda_H)^{-1}.
\end{aligned}$$

If $\lambda_N \le \lambda_H$,

$$\left\{ \begin{array}{ll}
(f_1^* \mid S_1 = s) = \infty & \text{if } (p_1 \mid S_1 = s)[c_H(t_2) - c_H(t_1)]^\beta\\
& \quad < (1 - (p_1 \mid S_1 = s))(c_N)^\beta\\
(f_1^* \mid S_1 = s) = 0 & \text{if } (p_1 \mid S_1 = s)[c_H(t_2) - c_H(t_1)]^\beta\\
& \quad \ge (1 - (p_1 \mid S_1 = s))(c_N)^\beta
\end{array} \right\}.$$

A waiting time of ∞ means that we wait until we get the next signal or until the user leaves the system. Note that when the model parameters are known, $(f_1^* \mid S_1 = 0)$ and $(f_1^* \mid S_1 = 1)$, and hence, $V_N(\hat{S}_0 \cup \{s\}, t_1)$ and $V_H(\hat{S}_0 \cup \{s\}, t_1)$ can be computed using the parameter values. Substituting these values into (4) and solving for f_0 gives the following solution:

If $\lambda_N > \lambda_H$,

$$f_0^* = \frac{\ln(\lambda_N(1-p_0)((c_N)^\beta - B)) - \ln(\lambda_H p_0 A)}{\lambda_N - \lambda_H}.$$

If $\lambda_N \le \lambda_H$,

$$\left\{\begin{array}{ll} f_0^* = \infty & \text{if } p_0 A + (1-p_0)B < (1-p_0)(c_N)^\beta \\ f_0^* = 0 & \text{otherwise} \end{array}\right\},$$

where $A = (1-\theta_1^H)([c_H(t_1) - c_H(t_0)] + C_H(\hat{S}_0 \cup \{0\}, t_1))^\beta + \theta_1^H([c_H(t_1) - c_H(t_0)] + C_H(\hat{S}_0 \cup \{1\}, t_1))^\beta$ and $B = \theta_1^N(C_N(\hat{S}_0 \cup \{0\}, t_1))^\beta + (1-\theta_1^N)(C_N(\hat{S}_0 \cup \{1\}, t_1))^\beta$.

In the above expressions, $(1-p_0)c_N^\beta$ represents the disutility of taking action immediately and $p_0 A + (1-p_0)B$ represents the disutility of waiting for the next signal. If the damage cost follows a stepwise function, then we can implement standard backward induction to derive the optimal waiting-time policy. However, analytical solutions are difficult to obtain for other cost functions. Because we desire an implementable policy, we seek approximate solutions that are easy to compute. One straightforward strategy is to approximate the damage cost using a stepwise function so that we can readily apply backward induction. A second approach is to solve a myopic model that minimizes only the disutility incurred until the next decision point. That is, at every decision point, we minimize the cost incurred up to the *next* decision point instead of considering *all* future decision points. We analyze this model next.

5. Myopic Policy

Under the myopic policy, at every decision point i, we determine f_i by minimizing the expected disutility incurred during the period between t_i and the time at which we make the next decision, which could be either t_{i+1} or $t_i + f_i$. Under this policy, disutility beyond $\min\{t_i + f_i, t_{i+1}\}$ is ignored. That is, the optimization model for the myopic policy can be derived from the exact model (3) by setting $C_N(\hat{S}_i \cup \{s\}, t_{i+1})$ and $C_H(\hat{S}_i \cup \{s\}, t_{i+1})$ to zero. This gives the following disutility function for the myopic policy:

$$\begin{aligned} &V(\hat{S}_i, t_i) \\ &= \min_{f_i} p_i \int_{t_i}^{t_i+f_i} \lambda_H e^{-\lambda_H(t_{i+1}-t_i)}[c_H(t_{i+1}) - c_H(t_i)]^\beta \, dt_{i+1} \\ &\quad + e^{-\lambda_H f_i} p_i [c_H(t_i + f_i) - c_H(t_i)]^\beta + e^{-\lambda_N f_i}(1-p_i)c_N^\beta. \end{aligned} \tag{5}$$

The first-order condition for the above model is

$$\begin{aligned} &p_i e^{-\lambda_H f_i} \beta [c_H(t_i + f_i) - c_H(t_i)]^{\beta-1} \left.\frac{\partial c_H}{\partial t}\right|_{t_i+f_i} \\ &\quad - \lambda_N e^{-\lambda_N f_i}(1-p_i)c_N^\beta = 0. \end{aligned} \tag{6}$$

The second-order condition for minimization is

$$\begin{aligned} &-\lambda_N(\lambda_H - \lambda_N)e^{(\lambda_H-\lambda_N)f_i}(1-p_i)c_N^\beta \\ &\quad + p_i\beta(\beta-1)[c_H(t_i+f_i) - c_H(t_i)]^{\beta-2}\left(\left.\frac{\partial c_H}{\partial t}\right|_{t_i+f_i}\right)^2 \\ &\quad + p_i\beta[c_H(t_i+f_i) - c_H(t_i)]^{\beta-1}\left.\frac{\partial^2 c_H}{\partial t^2}\right|_{t_i+f_i} > 0, \end{aligned}$$

which is satisfied when $\lambda_N \ge \lambda_H$. We assume $\lambda_N \ge \lambda_H$ in our subsequent analysis. Let the optimal waiting times at t_i under the optimal policy and the myopic policy be $f_i^{*\text{optimal}}$ and $f_i^{*\text{myopic}}$, respectively. Then, we can show the following result.

Proposition 2. *$f_i^{*\text{optimal}} \le f_i^{*\text{myopic}}$ for all i.*

Because the myopic policy ignores the disutility incurred beyond the next signal, it underestimates the cost of waiting. Consequently, the myopic policy results in a longer-than-optimal waiting time and, hence, a larger expected damage cost from hacking.

The myopic policy is the solution to the first-order differential equation in (6), which is simpler to solve than the optimal model for a variety of functional forms of c_H. For example, for $\lambda_H = \lambda_N = \lambda$ and the linear damage cost function, $c_H(t) = At$, and in case of risk neutrality, while the optimal policy cannot be derived in closed form, the myopic policy is as follows:

$$\left.\frac{\partial c_H}{\partial t}\right|_{t=t_i+f_i} = \lambda\left(\frac{1}{p_i} - 1\right)c_N, \tag{7}$$

which can be solved analytically for a variety of functional forms of c_H. For the risk-averse case, obtaining closed-form solutions for the myopic policy is more difficult; however, numerical solutions can be easily obtained. Further analysis of the waiting time under the myopic policy, characterized by (6), shows the following results about how various parameters affect the waiting time under the myopic policy.

Proposition 3 (Characteristics of the Myopic Policy). (i) $\partial f_i/\partial c_N > 0$, (ii) $\partial f_i/\partial p_i < 0$, (iii) $\partial f_i/\partial \lambda_H > 0$, (iv) *if* $f_i > 1/\lambda_N$ *then* $\partial f_i/\partial \lambda_N < 0$ *else* $\partial f_i/\partial \lambda_N > 0$, (v) *if* $([c_H(t_i+f_i) - c_H(t_i)]/c_N) < e^{(-1)/\beta}$ *then* $\partial f_i/\partial \beta > 0$ *else* $\partial f_i/\partial \beta < 0$, (vi) $\partial f_i/\partial t_i < 0$.

Because the myopic policy balances the cost of taking action against a normal user with the benefit of obtaining additional information about the user, a higher cost of taking action against a normal user

results in a longer waiting time as shown in Proposition 3(i). Proposition 3(ii) shows that at any decision point, a higher probability that the user is a hacker results in a shorter waiting time. The reason for this is that a higher probability decreases the expected benefit from waiting for additional information.

Intuitions for Propositions 3(iii) and 3(iv) are not as obvious as for Propositions 3(i) and 3(ii). Proposition 3(iii) states that a higher signal frequency from a hacker increases the waiting time. However, more frequent signals from a normal user increase the waiting time only when the mean inter-arrival time of signals from normal users is more than the waiting time (Proposition 3(iv)). These results are somewhat counterintuitive as we would expect more frequent signals, or smaller time intervals between signals, to reduce the waiting time. The explanation for these results lies in the comparison between the marginal benefit of waiting for the new information, given by $\lambda_N e^{-\lambda_N f_i}(1-p_i)c_N$, and the marginal loss from waiting due to possible higher damage, given by $p_i e^{-\lambda_H f_i}\partial c_H/\partial t|_{t_i+f_i}$, as in (6). At optimality, we choose the waiting time for which the marginal benefit from waiting equals the marginal loss from waiting. The marginal-cost and marginal-benefit expressions reveal that the arrival rate of hacker (normal user) signals affects only the marginal cost (benefit) from waiting. When the signals from a hacker are more (less) frequent, the marginal cost from waiting decreases (increases) as evident from the term $e^{-\lambda_H f_i}$, but the marginal benefit remains unaffected, which, in turn, causes an increase (decrease) in the waiting time. Another, perhaps more intuitive, explanation for why we will increase the waiting time when the arrival rate of signals from hackers increases is as follows. If hackers were to take actions more quickly, the waiting time to receive the next signal will be sooner. Extending the waiting time in this case is not likely to increase the loss from a hacker because we are likely to get a signal from the hacker sooner than before, and consequently, we will have an opportunity to make a more informed decision. However, extending the waiting time reduces the risk of terminating a normal user. On the contrary, reducing the waiting time when the signals from a hacker become more frequent will definitely increase the risk of terminating a normal user, but does not guarantee that that loss from a hacker will decrease because our opportunity to make a more informed decision is also reduced.

The marginal-benefit expression, $\lambda_N e^{-\lambda_N f_i}(1-p_i)c_N$, which depends only on the arrival rate of signals from a normal user, has the shape of an exponential distribution, as shown in Figure 3. When the arrival rate of information from a normal user increases from λ_1 to λ_2, the marginal benefit of waiting for the additional information increases only when $f_i < f^*$,

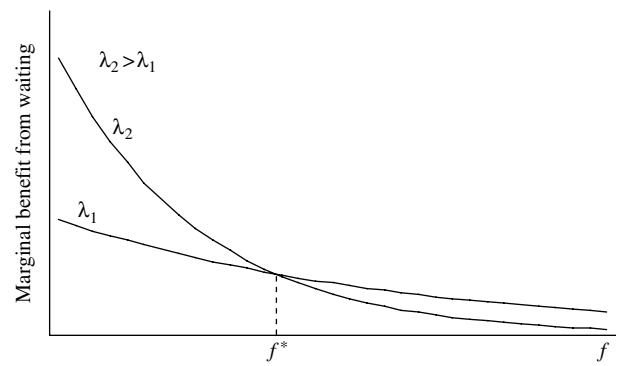

Figure 3 **Marginal Benefit from Waiting for Various Values of λ_N**

so Proposition 3(iv) results. Intuitively, if the signals from a normal user become more frequent, then waiting longer before taking action may not reduce the cost of incorrectly terminating a normal user. This occurs when the waiting time is already large enough so that the probability of incorrectly terminating a normal user is small. In this situation, we will be better off reducing the waiting time to at least reduce the possible damage from hackers. On the other hand, when the signals from a normal user become more frequent, if waiting to take action reduces the cost of incorrectly terminating a normal user, then we will benefit from increasing the waiting time. However, the increase in the waiting time will be mitigated by the increase in damage by hackers.

Proposition 3(v) shows the effect risk aversion on the waiting time. It is intuitive that when the cost of waiting, i.e., $c_H(t_i+f_i)-c_H(t_i)$, is more than the cost c_N of taking incorrect action, a higher risk aversion, i.e., higher β, will always result in a shorter waiting time. This follows from the fact that the right-hand side of the condition in Proposition 3(v), $e^{-1/\beta}$, is less than one. If the cost of waiting is less than the cost of taking an incorrect action, then the waiting time may initially increase with an increase in risk aversion, but begins to decrease when risk aversion is sufficiently high.

Proposition 3(vi) shows that the waiting time decreases with t, as expected because as t increases, the marginal cost of waiting increases. However, the benefit from waiting does not change with time. Consequently, waiting becomes less attractive and we decrease the waiting time as t increases.

The myopic policy is easier to implement than is the optimal policy. Implementing the myopic policy as a separate DSS tool is appropriate for passive IDSs that only give signals about user activities. It can also be embedded into IDSs to make them active so that they not only detect intrusions but also take actions.

While the myopic policy has nice theoretical properties as shown in Proposition 3, prior to implementing the myopic or any other policy, we need to determine how well the policy performs compared to

the optimal policy in the target environment. In the next section, we compare the optimal policy with the myopic and other common static policies via numerical experiments.

6. Comparison of Optimal, Myopic, and Other Policies

We performed two sets of numerical experiments. In the first, we investigated whether the behavior of the myopic policy is similar to that of the optimal policy, enabling us to conclude whether the myopic policy is a good substitute for the exact policy. In the second set of experiments, which was much more extensive, we compared the myopic policy with the simple Bayes policy and a family of static policies to which we refer as the m-strike policies. Further, in the theoretical analysis we assumed that all exogenous parameters are known constants. In the second set of numerical experiments, we relaxed this assumption and incorporated variability into the parameters.

6.1. Comparison of Optimal and Myopic Policies

We assumed that a user entered the system at time zero and remained in the system for two actions, leaving the system if the system did not terminate the user before two actions. We let $C_H(t) = At$. The linear damage cost function and the restriction of the number of user actions to two were required to derive the exact optimal policy. Further, we assumed risk neutrality.

We simulated 100 batches of 1,000 users each for each set of parameter values: $p_0 \in \{0.001, 0.01, 0.05, 0.1\}$, $\lambda_H, \lambda_N \in \{1, 2, 4, 8\}$, $\theta_H = \theta_N \in \{0.5, 0.75, 0.9, 0.99\}$, $c_N \in \{0.5, 1, 2, 5\}$, and $C_H(t) = At$, $A \in \{1, 2, 5, 10\}$. We restricted $\lambda_H \leq \lambda_N$. The firm's total cost per user (TC) was lower under the optimal policy than under the myopic policy. The difference between the costs under the optimal policy and the myopic policy ranged from 0% to 16% of optimal. Table A1 in the Online Supplement provides our estimates of total cost per user, depicted in Figures 4–9, which compare optimal and myopic policies and standard errors of these estimates (the standard errors are all sufficiently small to make our comparisons statistically significant). The effects of c_N on TC under the two waiting-time policies are in Figure 4. TC increases as c_N increases under both policies because, as the cost of taking action against a normal user increases, waiting for information becomes more attractive compared to taking an action. However, the additional waiting results in a higher damage cost due to hacking (Figure 5).

In Figure 6, a higher θ_H and θ_N represents a higher-quality IDS, and TC decreases under both policies. When the IDS is near perfect ($\theta_H = \theta_N = 0.99$), the time to detect a hacker is close to zero, and the accuracy of detection is close to 100%. In the near-perfect IDS case, the optimal policy and the myopic policy are almost identical.

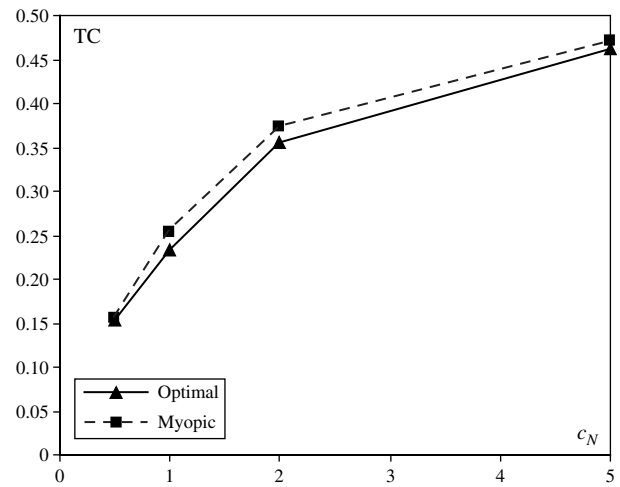

Figure 4 **Impact of** c_N ($\lambda_H = 2$, $\lambda_N = 2$, $A = 5$, $\theta_H = \theta_N = 0.75$, $p_0 = 0.01$)

The effect of the proportion of hackers in the user population is in Figure 7 while the effects of rates of information arrivals from users are shown in Figures 8 and 9. Under both policies, as the proportion of hackers increases, TC increases first, but TC starts to decrease when the proportion of hackers reaches a threshold value because, as the proportion of hackers increases, both policies reduce their waiting times terminating users faster, which increases the cost of mistermination. The cost increases initially, but after the proportion of hackers reaches a threshold, there is no need to shorten the waiting time; instead, we can eliminate the waiting time altogether. The effects of arrival rates of information from hackers and normal users are similar for both policies, and are consistent with Propositions 3(iii) and 3(iv). TC

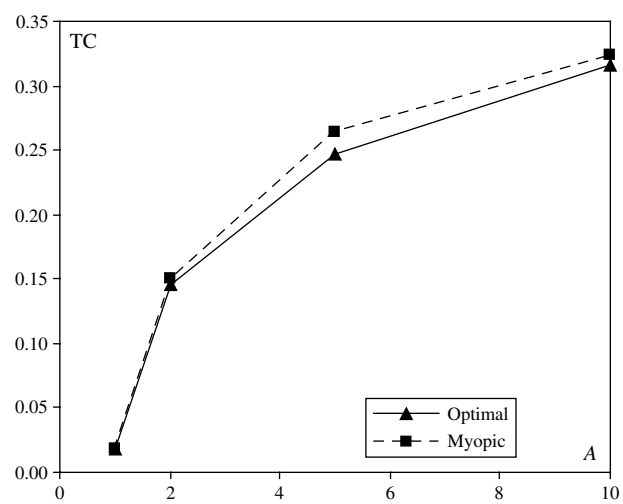

Figure 5 **Impact of** A ($\lambda_H = 2$, $\lambda_N = 2$, $c_N = 1$, $\theta_H = \theta_N = 0.75$, $p_0 = 0.01$)

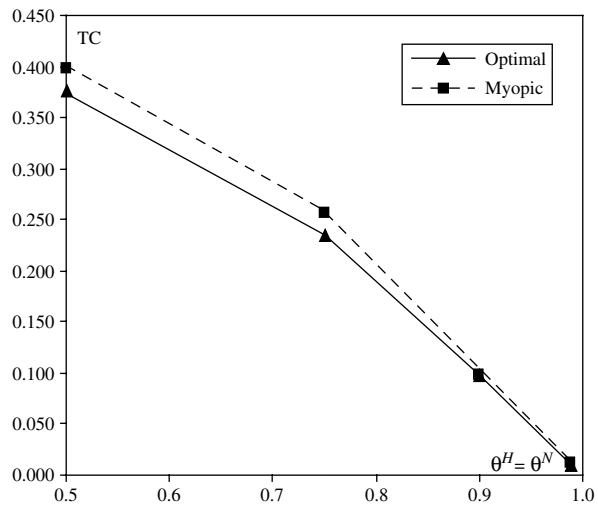

Figure 6 **Impact of IDS Quality** ($\lambda_H = 2$, $\lambda_N = 2$, $A = 5$, $c_N = 1$, $p_0 = 0.01$)

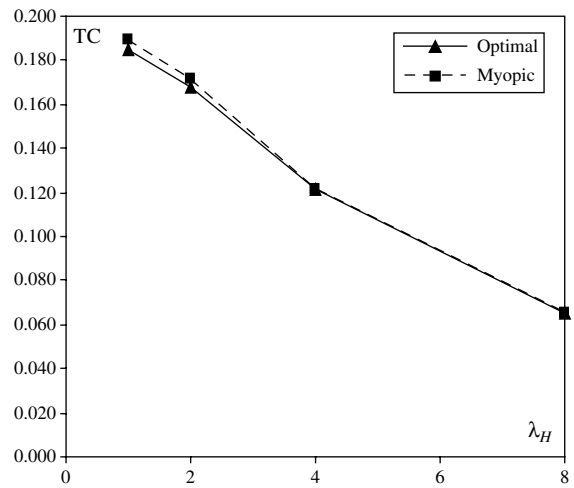

Figure 8 **Impact of** λ_H ($\lambda_N = 8$, $A = 5$, $c_N = 1$, $\theta_H = \theta_N = 0.75$, $p_0 = 0.01$)

decreases with λ_H since a higher λ_H allows us to improve the accuracy of detection. In our simulations, TC decreases with λ_N also.

6.2. Comparison of the Myopic Policy with Bayes and m-Strike Policies

In the second set of simulations, we removed many of the restrictions imposed in the first set. Specifically,

(a) We considered risk aversion as well, with a quadratic disutility function.

(b) We allowed the parameter values p_0, λ_H, λ_N, and A to come from normal distributions, with standard deviation equal to one fourth of the mean, because estimates of these parameters often have significant uncertainty. The mean values were the same as those used in the first set. We excluded values that were outside the valid range for a parameter, viz., negative values, from our simulations.

(c) We allowed a user to remain in the system for five actions instead of two.

The second set of experiments compares the myopic policy with other commonly used policies, specifically the Bayes and the m-strike.

In the Bayes policy, either we take action immediately after receiving a signal from the IDS, or wait to get another signal. Thus, if the expected cost of taking action immediately after a signal is smaller than the expected cost of waiting for another signal, we take action immediately after receiving the signal. This is a specialized version of the myopic policy in that f_i is restricted to either zero (immediate action) or infinity (waiting for the next signal). Unlike m-strike policies, the Bayes policy considers the firm's cost, and, unlike myopic policy, it does not compute optimal waiting time. Substituting $f_i = 0$ in (5), we get the expected disutility of taking an action immediately as $(1-p_i)c_N^{\beta}$.

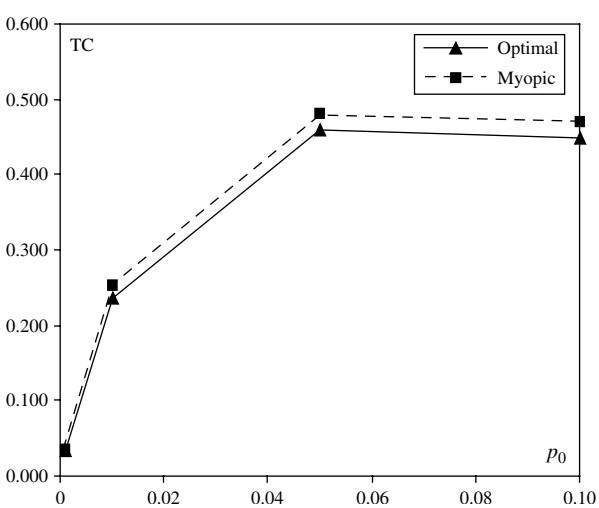

Figure 7 **Impact of Prior Probability** ($\lambda_H = 2$, $\lambda_N = 2$, $A = 5$, $c_N = 1$, $\theta_H = \theta_N = 0.75$)

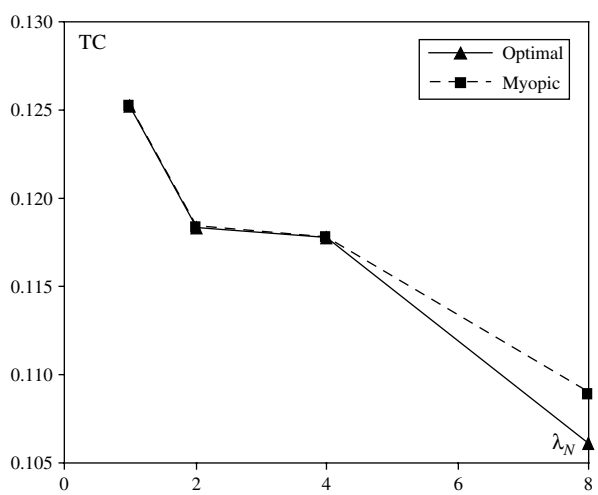

Figure 9 **Impact of** λ_N ($\lambda_H = 1$, $A = 5$, $c_N = 1$, $\theta_H = \theta_N = 0.75$, $p_0 = 0.01$)

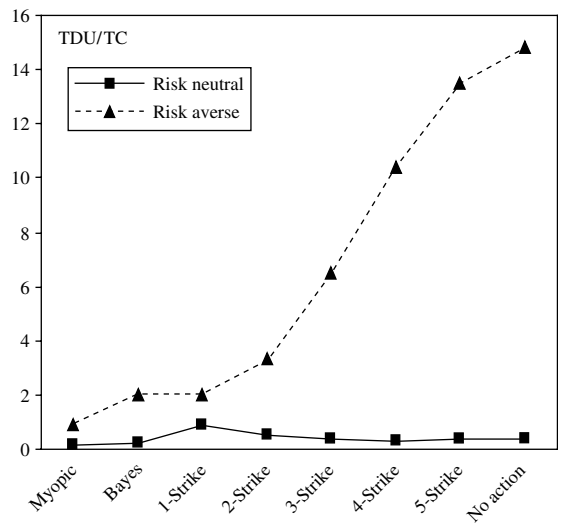

Figure 10 TDU (TC for Risk-Neutral Case)

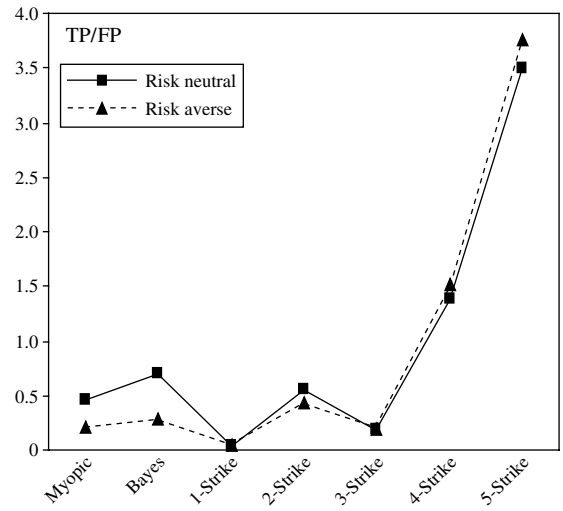

Figure 12 TP/FP for Various Policies

Letting $f_i \to \infty$ in (5), we get the expected disutility of waiting for the next signal as $\beta(A/\lambda_H)^\beta p_i$. Thus, the Bayes policy takes action at time t_i if $(1-p_i)c_N^\beta < \beta(A/\lambda_H)^\beta p_i$ given that $C_H(t) = (At)^\beta$.

The m-strike policy takes action when we receive m intrusion signals from the IDS. The no-action policy ignores IDS signals altogether, the same as the m-strike policy when $m \to \infty$. We consider these policies since the base-rate fallacy may cause us to ignore signals. The 1-strike trusts the IDS completely, and the ∞-strike policy completely distrusts IDS. m-strike policies do not minimize cost, but they may be attractive because they require no estimation cost parameters or probabilities, and can be easily implemented.

We compare these policies on expected total disutility (TDU), identical to the expected TC under risk-neutrality, expected time to detect a hacker (TD), true-positive-to-false-positive ratio (TP/FP), and our estimate of the probability that the user is a hacker when we receive m intrusion signals given that the user is in fact a hacker. The last measure is used to compare only m-strike policies.

We simulated 100 batches of 1,000 users each for each set of parameter values. We used the same combinations of parameter values as before, but now they are the means of normal distributions. The average values for each performance measure, computed over all scenarios generated for each policy, are in Figures 10–13. The simulation results (both mean and standard error) used to generate Figures 10–13 are provided in Tables S1–S6 of the Online Supplement.

The myopic policy had the least expected disutility for both the risk-neutral and risk-averse cases (Figure 10). The difference between the expected disutility under a given policy and that under the myopic policy, as percentage of the latter, is in Table 2. The

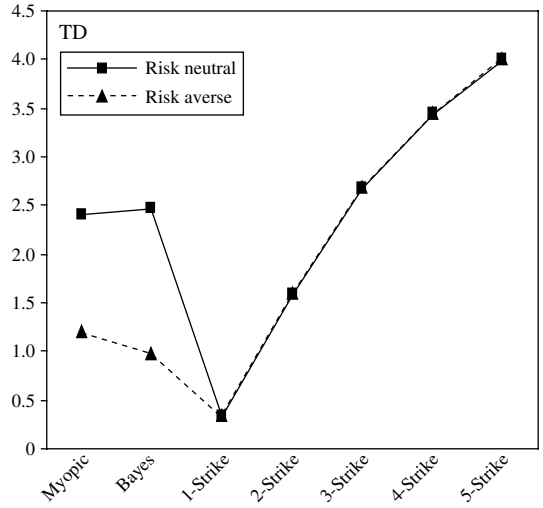

Figure 11 TD for Various Policies

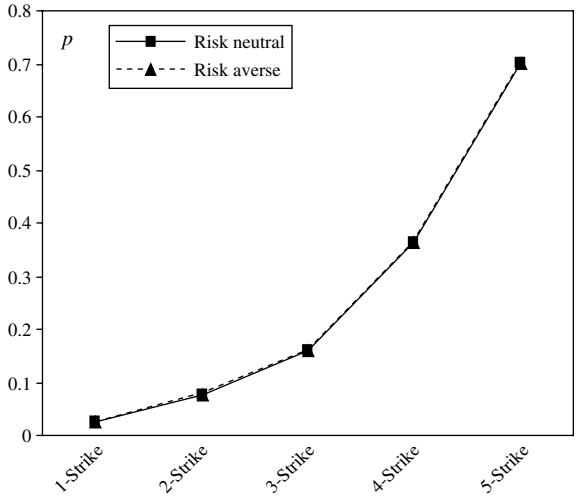

Figure 13 Probability that the User Is a Hacker Given the User Was Terminated

Table 2 Percent Increase in Disutility Relative to the Myopic Policy

Policy\% increase	0–50	50–200	200–500	>500
Bayes	57	30	9	4
1-Signal	11	37	30	22
2-Signal	0	57	21	21
3-Signal	28	34	15	23
4-Signal	33	18	13	36
5-Signal	38	15	13	35
No Action	35	17	13	35

Bayes and static policies result in more than a 50% increase in disutility in more than 43% of the trials. The myopic clearly beats other simpler and static policies.

While the time to detect a hacker is not the lowest and the TP/FP ratio is not the highest under the myopic policy, the myopic policy trades off these two quantities to achieve a lower average cost. In Figure 13, for a hacker, the estimate of the probability that the user is a hacker given m intrusion signals increases with m, so given a hacker, additional waiting enabled by a larger m improves the estimate of the probability that the user is a hacker; i.e., waiting mitigates the base-rate fallacy. Within the family of m-strike policies, detection accuracy improves with m, but expected cost also increases, implying that high accuracy does not translate into an overall positive effect if achieving higher accuracy takes longer, because damage cost increases with time. The optimal policy trades off higher accuracy and time required to achieve it.

In summary, the optimal policy was qualitatively similar to the myopic policy, so the theoretical results on the myopic policy are likely to hold for the optimal policy too. The myopic policy is nearly identical to the optimal policy when the cost of taking action against a normal user is high, the cost of undetected hacking does not increase significantly with time, the quality of the IDS is either sufficiently high or sufficiently low, the proportion of hackers in the user population is low, or when hackers (normal users) take more (fewer) actions per unit time. The myopic policy was better than Bayes and m-strike in almost all cases.

7. Conclusions and Limitations

Implementation of the proposed policies has several limitations. Both myopic and optimal policies require several parameters related to costs, the IDS, and hacker behavior. Though estimates of security costs are difficult to obtain, there are efforts to quantify these costs (Lee et al. 2001, Iheagwara 2004, Cavusoglu et al. 2004). Gordon and Loeb (2001) state that security frauds are often committed to gather competitive intelligence data. Thus the dollar amount the competitors are willing to pay for such information can be used to estimate the loss from security breaches. There is a vast literature on evaluation, benchmarking, and comparison of security products (Aguirre and Hill 1997, Puketza et al. 1997, Lippmann et al. 2000, Durst et al. 1999, NSS Group 2001). Obtaining user-related data, for example, the benefit from hacking is more difficult. Data collection has begun about hacker behavior with the help of honeypots (Honeynet Project 2004, Spitzner 2002). If good estimates are unavailable, our model can still be used to perform sensitivity analysis. Further, we used a decision-theoretic approach in that we ignored the hacker's reactions to the waiting-time policies. Given that IT security can be viewed as a game between the firm and hackers, incorporating game-theoretic aspects could enrich the model.

One potentially fruitful extension lies in modeling the information content of the signal from IDSs. We assumed that the IDS generates a binary signal indicating whether the user's activity is intrusive. Because many IDSs serve as decision support tools, developers have begun to enrich the signal by incorporating additional information such as attack severity and likelihood of attack in these signals. We assumed that the firm uses only one IDS, but multiple IDSs with different capabilities could increase detection accuracy. While higher detection accuracy mitigates the base-rate problem, determining optimal waiting time remains as long as detection is imperfect. Another extension pertains to handling multiple attack types with different levels of damage, requiring estimation of parameters related to different attack types, their relative frequencies, and the damage caused by each.

Acknowledgments

An earlier version of this paper was presented at the 2003 Workshop on Information Technology and Systems (WITS) in Seattle, WA, where it received the best paper award. The authors thank the reviewers of and participants in the WITS workshop, and the seminar participants at the University of Texas at Dallas for valuable comments.

References

Aguirre, S. J., W. H. Hill. 1997. Intrusion detection fly-off: Implications for the United States Navy. MITRE Technical Report MTR 97W096, MITRE, McLean, VA.

Axellson, S. 2000. The base-rate fallacy and the difficulty of intrusion detection. *ACM Trans. Inform. System Security* **3** 186–205.

Cavusoglu, H., S. Raghunathan. 2004. Configuration of detection software: A comparison of decision and game theory approaches. *Decision Anal.* **1** 131–148.

Cavusoglu, H., B. Mishra, S. Raghunathan. 2004. The effect of internet security breach announcements on market value: Capital market reaction for breached firms and internet security developers. *Internat. J. Electronic Commerce* **9** 69–105.

Cavusoglu, H., B. Mishra, S. Raghunathan. 2005. The value of intrusion detection systems (IDSs) in information technology (IT) security. *Inform. Systems Res.* **16** 28–46.

Denning, D. E. 1987. An intrusion detection model. *IEEE Trans. Software Engrg.* **13** 222–232.

D'haeseleer, P., S. Forrest, P. Helman. 1996. An immunological approach to change detection: Algorithms, analysis, and implications. *IEEE Sympos. Security and Privacy*. IEEE Press, New York.

Diamond, H. 1982. Minimax policies for unobservable inspections. *Math. Oper. Res.* **7** 139–153.

Durst, R., T. Champion, B. Witten, E. Miller, L. Spagnuolo. 1999. Testing and evaluating computer intrusion detection systems. *Comm. ACM* **42** 53–61.

Escamilla, T. 1998. *Intrusion Detection: Network Security Beyond the Firewall*. John Wiley & Sons, New York.

Gartner. 2003. Hype cycle for information security. Gartner Research Report, Gartner, Stamford, CT.

Gordon, L. A., M. P. Loeb. 2001. Using information security as a response to competitor analysis systems. *Comm. ACM* **44** 70–75.

Honeynet Project. 2004. *Know Your Enemy: Learning about Security Threats*. Addison-Wesley, Boston.

Iheagwara, C. 2004. The effect of intrusion detection management methods on the return on investment. *Comput. Security* **23** 213–228.

Jonsson, E., T. Olovsson. 1997. A quantitative model of the security intrusion process based on attacker behavior. *IEEE Trans. Software Engrg.* **23** 235–245.

Kumar, S., E. Spafford. 1996. A pattern matching model for misuse intrusion detection. *The COAST Project*. Purdue University, West Lafayette, IN.

Lee, W., W. Fan, M. Miller, S. Stolfo, E. Zadok. 2001. Toward cost-sensitive modeling for intrusion detection and response. *J. Comput. Security* **10** 5–22.

Lippmann, R. P., D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, S. E. Weber, D. Webster, R. K. Wyschogrod, R. K. Cunningham, M. A. Zissman. 2000. Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. *Proc. 2000 DARPA Inform. Survivability Conf. Exposition*, IEEE Press, Los Alamitos, CA, 12–26.

Lunt, T. 1993. A survey of intrusion detection systems. *Comput. Security* **12** 405–418.

Moitra, S., S. Konda. 2000. A simulation model for managing survivability of networked information systems. Technical Report, Carnegie Mellon Software Engineering Institute, Carnegie Mellon University, Pittsburgh.

Neumann, P., P. Porras. 1999. Experience with emerald to date. *Proc. First USENIX Workshop on Intrusion Detection and Network Monitoring, Santa Clara, CA*, 73–80.

NSS Group. 2001. *Intrusion Detection Systems Group Test*, 2nd ed. Oakwood House, Wennington, Cambridgeshire, UK.

Ozekici, S., S. Pliska. 1991. Optimal scheduling of inspections: A delayed Markov model with false positives and negatives. *Oper. Res.* **39** 261–273.

Porras, P., R. Kemmerer. 1992. Penetration state transition analysis: A rule based intrusion detection approach. *IEEE Eighth Annual Comput. Security Appl. Conf.*, IEEE Press, Los Alamitos, CA, 220–229.

Porras, P., P. Neumann. 1997. Emerald: Event monitoring enabling responses to anomalous live disturbances. *Proc. 20th National Inform. Systems Security Conf.*, National Institute of Standards and Technology, Baltimore, 353–365.

Puketza, N., M. Chung, R. O. Olsson, B. Mukherjee. 1997. A software platform for testing intrusion detection systems. *IEEE Software* **14** 43–51.

Ross, S. 1983. *Introduction to Stochastic Dynamic Programming*. Academic Press, New York.

Russell, D., G. T. Gangemi. 1992. *Computer Security Basics*. O'Reilly & Associates, Inc., Sebastopol, CA.

Shipley, G. 1999. ISS realsecure pushes past newer IDS players. *Network Comput.* **10** 95–111.

Spitzner, L. 2002. *Honeypots: Tracking Hackers*. Addison-Wesley, Boston.

Ulvila, J., J. Gaffney. 2004. A decision analysis method for evaluating computer intrusion detection systems. *Decision Anal.* **1** 35–50.

Zamboni, D., E. Spafford. 1999. New directions for the AAPHID architecture. *Recent Advances in Intrusion Detection*. Purdue University, West Lafayette, IN.