

# A Cost-Sensitive Model for Preemptive Intrusion Response Systems

Natalia Stakhanova

Samik Basu

Johnny Wong

Department of Computer Science  
Iowa State University  
Ames, IA 50011 USA  
{*ndubrov, sbasu, wong*}@iastate.edu

## Abstract

*The proliferation of complex and fast-spreading intrusions not only requires advances in intrusion detection mechanisms but also demands development of sophisticated and automated intrusion response systems. In this paper we present a novel cost-sensitive model for intrusion response that incorporates preemptive deployment of the response actions. Specifically, our technique relies on comparing the cost of deploying a response against the cost of damage caused by an “un-attended” intrusion and decides to preemptively deploy a response with maximum benefit. Our technique further allows adaptation of responses to the changing environment through evaluation of success and failure of previously triggered responses. We demonstrate the advantages of the approach and evaluate it using a damage reduction metric.*

## 1 Introduction

Intrusion detection has been at the center of intense research in the last decade owing to the rapid increase of sophisticated attacks on computer systems. Typically, intrusion detection refers to a variety of techniques for detecting attacks in the form of malicious and unauthorized activity. In the event that an intrusive behavior is detected, it is desirable to take (evasive and/or corrective) actions to thwart attacks and ensure safety of the computing environment. Such counter-measures are referred to as *intrusion response*. Although the intrusion response component is often integrated with the intrusion detection system (IDS), it receives considerably less attention than IDS research owing to the inherent complexity in developing and deploying response in an automated fashion.

**Driving Problem.** Traditionally, triggering an intrusion response is left as part of the administrator’s responsibility

requiring a high-degree of expertise. Although in the recent years this focus has shifted, automatic intrusion response support nowadays is still very limited. Most of the automatic response systems rely on the mapping of attacks to pre-defined responses [20, 3]. These approaches allow the system administrators to deal with intrusions faster and more efficiently. However, they lack flexibility mainly because few of these systems take into account intrusion cost factors. Recent years have seen increased interest in developing cost-sensitive modeling of response selection [19]. The primary aim for applying such a model is to realize a balance between intrusion damage and response cost to ensure adequate response without sacrificing the normal functionality of the system under attack. However, defining accurate measurement of these cost and risk factors is one of the challenges in using these models.

Another aspect that defines the efficiency of the intrusion response is the timeliness of the response deployment. The response action in the majority of the existing response systems is conservatively invoked once the existence of intrusion is confirmed. Though this strategy reduces false-positive response, delayed response action can potentially expose systems to higher level of risk from intrusions, specifically in cases where it becomes impossible to restore systems to its pre-attacked state.

Finally, the response mechanism must allow adaptation in response selection. Specifically, if a triggered response fails to handle the corresponding intrusion multiple times then it can be inferred that the system configuration, under which the triggered response was mapped to the intrusion, has changed; this requires adaptation of response-mappings.

**Our Solution.** In this paper, we present an intrusion response model which is

1. *automated*: deploying responses with little or no user-guidance.

2. *cost-sensitive*: assessing the risk and cost of (not) responding.
3. *pre-emptive*: triggering responses before the attack completes.
4. *adaptive*: updating the responses on-the-fly on the basis of their success and failure in thwarting previously seen intrusions.

Our intrusion response technique relies on the existence of a pattern-based intrusion detection framework. Such intrusion detection system can employ repositories of normal and anomalous (intrusive) patterns in a form of graphs [18], specify patterns of anomalous activities in a form of rules [7] or manually describe behavioral patterns using a specification language [15, 21]. We will specifically focus on IDS based on state-transition representation of system behavior [18]. The behavioral patterns are combined into state-transition graphs where states are represented by actions (system calls) that constitute the pattern. Detection is done by matching monitored sequence with patterns either in normal or abnormal graphs. If the monitored sequence exists in the normal graph, it is allowed to execute; otherwise if the sequence exists in abnormal graph, the sequence is classified as intrusive behavior.

We associate response action(s) with each intrusive pattern recorded in the abnormal graph; this is done by associating the start state of the pattern to the response(s). Monitored system activity is then continuously matched with patterns in normal and abnormal graphs. Whenever a monitored sequence matches with a prefix of any intrusive behavior (in abnormal graph), our algorithm decides whether or not to respond immediately. Note that, the monitored activity is not yet classified as intrusive – it just matches with a prefix of one or more intrusive behavior. If a response action is invoked at this stage, we are forcing *pre-emptive response* to a *possible* intrusive behavior. However, we do not allow *blind* pre-emption, instead pre-emption depends on the probability of the potential intrusion. If the prefix-pattern matched with the monitored sequence exhibits high probability (greater than a pre-specified threshold or tolerance factor) of ending up in an intrusive behavior then our algorithm decides to deploy a response action.

As a number of intrusive patterns can have the same prefix, there can be multiple candidate response actions that can be deployed pre-emptively. Therefore, it is required to identify the response that will have the *least negative affect* on the system. This is done by evaluating costs and benefits of the available responses based on the *utility theory* [4], that focuses on providing maximum benefit at the lowest risk. In particular, we take into consideration the past effectiveness of the possible responses and their respective severity. By severity, we imply the negative impact the response can have on legitimate users.

In the event intrusive behavior continues after the pre-emptive response was deployed, the triggered response’s effectiveness factor is lowered to *adjust response selection* in the future use of this response action. This response selection and deployment are performed automatically without any user intervention which allows fast containment of the intrusion and thus makes system defense more effective.

The paper’s main contributions lie in the following areas:

1. algorithm for determining preemptive deployment of the response.
2. the cost-sensitive methodology to intrusion response selection that is based on economic incentive and incorporates damage incurred by an attack and cost of responding to it.
3. the structured and comprehensive approach to automatic, preemptive, cost-sensitive and adaptable response.

**Organization.** The reminder of the paper is organized as follows. A brief overview of related work is given in Section 2. Section 3 presents the details of the response mechanism components. Experimental setup and results are given in Section 4. Section 5 concludes the paper with our future work.

## 2 Related Work

A number of techniques aiming at enhancing automation in the intrusion response were proposed and deployed over the last five years. Majority of them provide a pre-defined response actions mapped to pre-specified intrusion alerts [17, 15, 9]. Deployment of responses in these cases uses techniques ranging from static mapping table [17, 14, 2, 21] to dynamic rule-based selection procedure [12, 22].

Recently there have been a trend towards a self-healing systems that automatically find, diagnose, and respond to failures. While analysis technique varies from the application of checkpoints [13, 6] to the use of emulation environment [16], response component of these systems usually falls back into pre-determined set of actions. A number of techniques also proposed cost-sensitive analysis for selection of response actions [20, 8, 5]. An overview of these approaches is presented in [19]. As our approach employs cost-sensitive modeling technique, we will primarily focus on the related work in this area.

Models proposed by Toth and Kruegel [20] and Balepin et al. [1] consider costs and benefits of the response actions in association with dependencies between services in the system. Such modeling reveals priorities in response targets and evaluates the impact of different response strategies on dependent services and system.

The cost-sensitive approaches to intrusion response proposed by Lee et al. [8] and Foo et al. [5] attempt to balance intrusion damage and response cost. While [8] considers cost factors to improve intrusion detection, [5] specifically focuses on the containment of the attack.

The approach proposed in [5], called Adaptive Intrusion Response using Attack Graphs (ADEPTS), models intrusions using graph which allows to identify possible attack targets and consequently shows objectives of suitable responses. The response actions for the affected nodes in the graph are selected based on the effectiveness of this response to the particular attack in the past, the disruptiveness of the response to legitimate users and the probability measure that a real intrusion is taking place.

While our technique closely relates to ADEPTS, there are several distinguishing features. Our technique concentrates on the damage incurred by an attack, when ADEPTS considers responses from the degradation of system services perspective. ADEPTS approach relies on semi-manual development of *Intrusion-graph* to determine the spread of network attacks. The graph is static and acts as an auxiliary information used in conjunction to the actual intrusion detection system. If the computing environment changes, the I-graph needs to be updated accordingly using expert guidance. On the other hand, the graphical structure, our technique uses, records the attack patterns and is an integral constituent of the intrusion detection system; as such this structure can be dynamically and automatically updated with the introduction or classification of new attacks.

Another important difference in the two techniques comes from the response selection perspective. While both techniques use cost-sensitive approach ADEPTS picks the response action with the highest difference in response effectiveness and disruptiveness in normal activity, without taking into consideration attack damage and possibility that it may or may not exceed benefits of deploying a response. In contrast, our approach attempts to balance intrusion damage cost and response cost, and bases response selection on *utility theory* that allows to apply different security policies without any modifications of existing response metrics.

### 3 Intrusion Response Model

We associate a response action with each known pattern of anomaly. This can be done manually when patterns are stored in the corresponding specification repository or through the mapping to an evolving response taxonomy. Automating this process is difficult (if not impossible) and providing semi-automatic solution to response mapping is one of the future avenues of our research.

The deployment of the response is determined through the three-step process outlined in Figure 1. The goal of the first step is to determine when pre-emption should be in-

- 1: *determine when to start response selection:*  
     there exists sequence such that  
     confidence level  $\geq$  probabilityThreshold
- 2: *determine whether response action should be taken at this point:*  
     select responses such that  
     damageCost\*confidence level > responseCost
- 3: *select optimal response:*  
     compute expected value of attack  
     corresponding to selected responses

**Figure 1.** Algorithm for response deployment process.

voked. This depends on a pre-specified threshold (can be viewed as a tolerance level). We find whether the prefix of the sequence has reached a tolerable level or, in other words, a point where we are almost sure to see an actual attack.

In the second step, the set of candidate responses is identified, which if deployed at this point is going to cause less harm (due to incorrect pre-emption) than the damage caused by the possible intrusion. The selection of responses is based on the evaluation of *damage cost* and *response cost* factors. While *response cost* represents an effect of a response action on a system, *damage cost* generally quantifies the number of resources or computing power that can be left unusable by the attack. Setting accurate measurement of these cost factors is one of the challenges in using cost-sensitive models. As numeric values such as monetary values or percentages that correspond to some objective metrics are not always suitable, more effective solution based on relative measurements can be applied [11]. The relative measurements can be constructed based on company-specific security policies, existing threats, risk factors etc. [8]. The important factor that guides the cost values is the system mission. The costs should correspond to the level of disruption the attack might cause to the system. While it is difficult to determine ahead of time what exactly an attack will do, it is possible to recognize at least a direction of intrusion using pattern-based intrusion detection system. In practice, setting measurements of these costs is the responsibility of system administrator and is usually done manually. One of the downsides of this approach is the necessity to update cost factor values with time. In our approach, each attack pattern is associated with a damage cost using the prior information, in particular costs developed by [8].

In the final step, the response that has the *highest* chance of blocking a *possible* intrusion is selected and deployed from the candidate set. The details of this response selection process is presented below.

**Step 1: Preemption.** A response can be fired *eagerly* as soon as an incoming sequence is recognized as a (potential)

prefix of a known attack sequence. Note that at these early stages, the incoming sequence can be matched as a sub-sequence of multiple intrusive patterns (i.e. the incoming sequence can be associated with multiple responses). The response can be also fired *lazily*, i.e. after confirmation of intrusion. Lazy reaction to intrusion will associate and fire accurate response to an intrusion but can be risky as it allows more time to exploit a vulnerable system. On the other hand, eager response provides preemption but is aggravated by high probability of mistake.

To guide a response deployment process we define a `probability threshold` that indicates an acceptable level of confidence that some attack is in progress and a response action should be triggered. Once the probability of occurrence of a particular sequence, given its prefix, exceeds the pre-specified `probability threshold`, it can be inferred that an attack is imminent. Formally, probability of occurrence of a sequence, referred to as *confidence level* is defined as:

$$\frac{\text{number Of Sequence-Occurrences}}{\text{total Number Of Sequences With This Prefix}} \quad (1)$$

### Step 2: Cost-sensitive selection of possible responses.

Once we have decided to respond, i.e., probability of intrusion has exceeded the tolerance limit defined by `probability threshold`, we need to identify the possible responses that can be deployed. As alluded before, preemption in response deployment may lead to damage to the computing environment due to incorrect response selection or due to response deployment to false alarm. As such, our goal is to balance the preemptiveness of the response action with the accuracy of intrusion prediction. In other words, care is taken to ensure that the damage due to possible incorrect deployment of response does not overshadow the benefits of early response. This is realized by considering cost-sensitive approach using the following parameters, *damage cost (DC)* and *response cost (RC)*. Among the responses available for the intrusive sequences whose prefix matches the monitored pattern, we select response actions for which the following condition holds:

$$DC * \text{confidenceLevel} > RC \quad (2)$$

**Step 3: Cost-sensitive response invocation.** To determine the optimal among the response-actions selected in Step 2 we take into account two factors: (a) *Success Factor (SF)* and (b) *Risk Factor (RF)*. While the former is the percentage of times a response, under consideration, has succeeded in the past, the latter determines the severity of the response. By severity we imply the effect of the response on the resources and the legitimate users. A more severe response (i.e. with high RF value) is likely to stop the intrusion but it is also plagued by the disadvantage of adversely affecting system performance and usage. Essentially, *Risk Factor*

represents the *response cost* associated with a response action.

Optimal response action should provide the maximum benefit at the lowest risk. Such response strategy can be computed using utility theory [4] where *expected value (EV)* of response  $r_S$  to a sequence  $S$  can be defined as follows:

$$EV(r_S) = (Pr_{\text{succ}}(S) * SF) + (Pr_{\text{risk}}(S) * (-RF)) \quad (3)$$

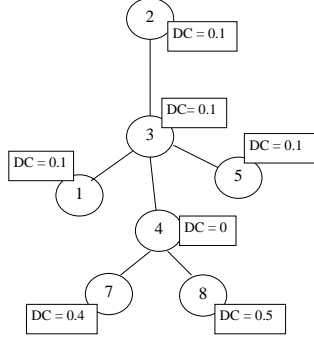
In the above  $Pr_{\text{succ}}(S)$  is the probability that a sequence  $S$  will occur and  $Pr_{\text{risk}}(S) = 1 - Pr_{\text{succ}}(S)$ . The optimal response action is selected based on the highest *EV*.

Note that the deployed response action might not correspond to the attack sequence whose confidence level exceeded a `probability threshold` in Step 1. Although this pattern is more likely to occur it might not provide optimal solution in terms of either damage cost being low or response cost being high. In either case, if the invoked response action fails to stop an intrusion, which is indicated by the continuance of the corresponding intrusive sequence, the response selection process will be triggered again. This time with increased confidence level of the correct intrusive pattern.

The primary advantage of utility theory-based approach is in its flexibility. The *RF* values can be adjusted according to one's preference (employing utility function [4]). In security-critical systems where exposure to external intrusion is more risky than possible inconvenience of users, we might choose to assign lower values to *RF*, thus decreasing the weights on  $Pr_{\text{risk}}(S)$ . In other systems, we might prefer to avoid any additional response risk even at the cost of losing information to the intruder (assign higher values to *RF*). Employing utility function also allows application of different security policies to the same set of response actions depending on the current system state. As such a high threat level can imply a use of utility function that adjusts weights of *RF* allowing deployment of more aggressive responses.

## 3.1 Adaptability

We consider the adaptability of the response mechanism in terms of its ability to adjust to the changing environment according to the response decisions previously issued by the system, in particular, success and failure of the triggered before responses. In the event, the selected response succeeds in blocking the attack, its *Success Factor* is automatically increased by one, otherwise, if the response fails to stop or divert a potentially anomalous pattern, its *SF* is decreased by one to reflect this result. Note that the update of the *SF* in case of failed response is performed after the monitored sequence fully matched the corresponding anomalous pattern to exclude the possibility of an error. This approach al-



(a) Example setting

Sequence	RF	SF	# of repetitions of each attack-pattern
2, 3, 1	0.9	1.0	2
2, 3, 5	0.5	0.8	1
2, 3, 4, 8	0.4	0.7	2
2, 3, 4, 7	0.7	0.8	1

Pattern seen:  $\langle 2 \rangle$  or  $\langle 2, 3 \rangle$

Sequence	RF	SF	Confidence level
2, 3, 1	0.9	1.0	(2/6) 0.33
2, 3, 5	0.5	0.8	(1/6) 0.16
2, 3, 4, 8	0.4	0.7	(2/6) 0.33
2, 3, 4, 7	0.7	0.8	(1/6) 0.16

(b) Steps 1 & 2

Pattern seen:  $\langle 2, 3, 4 \rangle$

Sequence	RF	SF	Confidence level
2, 3, 4, 8	0.4	0.7	<b>(2/3) 0.66</b>
2, 3, 4, 7	0.7	0.8	(1/3) 0.33

(c) Step 3

Select the responses

Sequence	RF	SF	Confidence level	DC	DC vs. RF
2, 3, 4, 8	0.4	0.7	(2/3) 0.66	0.7	$(0.7 * 0.66) > 0.4$
2, 3, 4, 7	0.7	0.8	(1/3) 0.33	0.6	$(0.6 * 0.33) < 0.7 \rightarrow$ <b>do not respond</b>

(d) Step 4

Deploy the best response

Sequence	RF	SF	Confidence level	EV
2, 3, 4, 8	0.4	0.7	(2/3) 0.66	$0.66 * 0.7 + (1 - 0.66) * 0.4 = 0.326$

(e) Step 5

**Figure 2.** Example demonstrating the response selection process

lows to adjust response selection for the future occurrences of this sequence and consequently, effectively adapts our response system to the changing environment.

### 3.2 Illustrative example

Let sequences presented in Figure 2(a) together with their graphical representation be a part of specifications representing anomalous patterns. Assuming that complete monitored pattern is  $\langle 2, 3, 4, 8 \rangle$  and probability threshold = 0.5 we determine the response action following the steps defined in Figure 1:

1. **Pattern observed:**  $\langle 2 \rangle$ . We compute confidence level for all sequences starting with prefix 2 (Figure 2(b)). Since none of the considered sequences has confidence level exceeding probability threshold we continue to monitor system executions without taking any action.
2. **Pattern observed:**  $\langle 2, 3 \rangle$ . Now confidence level is computed for sequences starting with prefix  $\langle 2, 3 \rangle$  (Figure 2(b)). Similarly, confidence

level of the considered sequences has not reached probability threshold.

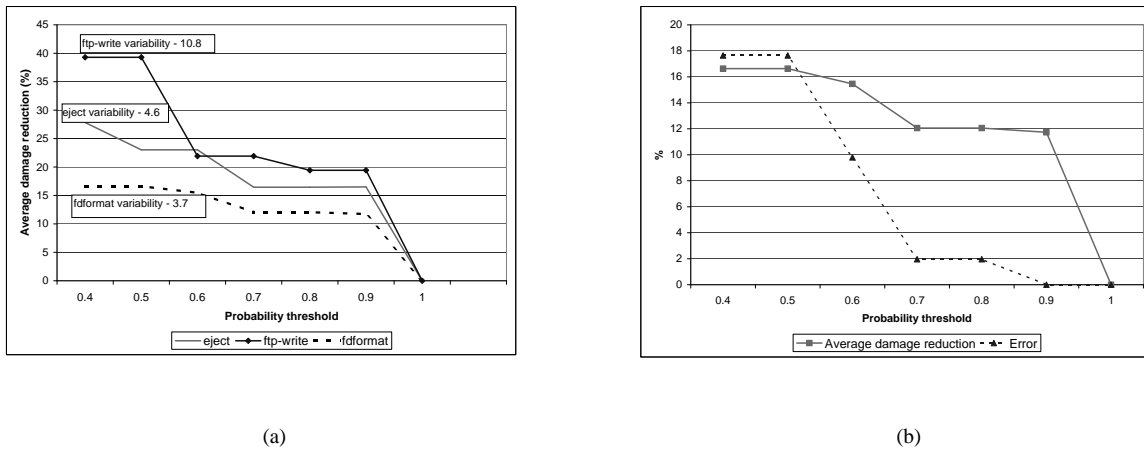
3. **Pattern observed:**  $\langle 2, 3, 4 \rangle$ . Now confidence level is computed for only two sequences starting with prefix  $\langle 2, 3, 4 \rangle$  (Figure 2(c)). Since confidence level  $>$  probability threshold, we select response actions that follow the condition (2) (Figure 2(d)). Since sequence  $\langle 2, 3, 4, 7 \rangle$  does not comply with this requirement which means deploying its response may cause more damage, it is eliminated from the further consideration. We compute EV value for the other sequence (Figure 2(e)). Since only one response action is left in this example, the last step is redundant, however, we provide the computations to demonstrate the entire response selection process. Finally, a response action associated with the sequence  $\langle 2, 3, 4, 8 \rangle$  is deployed.

## 4 Experiments

**Data.** We evaluated our model using the 1998 DARPA/Lincoln Lab offline evaluation data. We used Ba-

Attack	Description	Number of instances	Costs
<i>Eject attack</i>	exploits a buffer overflow vulnerability in 'eject' program.	16	DC = 1.0 RF = 0.4
<i>Fdformat attack</i>	exploits a buffer overflow using the 'fdformat' UNIX system command.	5	DC = 1.0 RF = 0.4
<i>Ftp-write attack</i>	exploits FTP server misconfiguration. Remote FTP user creates .rhosts file in world writable anonymous FTP directory and obtains local login.	2	DC = 0.5 RF = 0.2

**Table 1.** Attack descriptions [10].



**Figure 3.** (a) Average damage reduction. (b) Average damage reduction vs error (fdformat attack).

sic Security Module (BSM) audit records representing system calls and corresponding network traffic data indicating the attack starting points. For our experiments we used two User-to-Root attacks that exploit buffer overflows: *eject* and *fdformat* and one Remote-to-Local attack: *ftp-write*. The descriptions of these attacks and corresponding damage and response costs according to the attack taxonomy by [8] are given in Table 1. Each state in an attack trace is associated with a damage cost; the overall damage cost of the trace being the sum of damage costs of the states in the trace. Although our model allows associating multiple response actions with one anomalous sequence, for evaluation purposes we experimented with one response per sequence.

**Results.** Since the primary goal of this work is the intrusion response model we assume that repository of anomalous patterns is provided by the intrusion detection framework. We performed several experiments focusing on the effect of the cost-sensitive modeling and the preemptiveness of the response in our model. As a primary criteria we defined *damage reduction* metric that shows the difference between damage cost incurred by a full attack and a damage cost caused by the prefix of the attack sequence (at the time of the response). These measurements will be used to compare the systems without the cost-sensitive modeling with

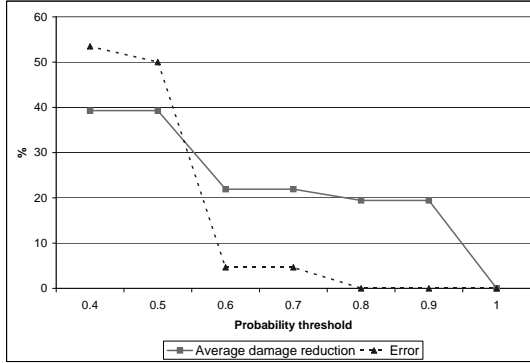
the one equipped with our response approach.

Figure 3(a) shows the average damage reduction for *eject*, *fdformat* and *ftp-write* attacks with respect to the probability threshold. The most significant damage reduction for all attacks occurs with the lowest value of probability threshold (0.4). A high value of probability threshold forces system to require more confirmation of the attack occurrence before a response can be deployed and, consequently, may increase the damage incurred by the late response. As such probability threshold = 1.0 does not carry any damage reduction and thus corresponds to the “cost-insensitive” system.

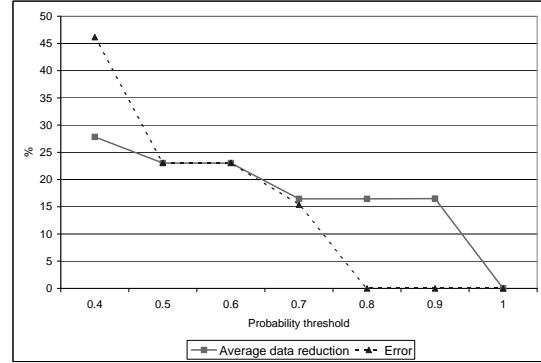
It is important to note that all considered attacks have similar damage reduction patterns with only curve difference. The difference in the graph can be correlated to the average variability in attack-patterns with identical prefix. The variability is computed as the weighted mean of the *prefixLength* valuations where the weights are defined by the frequency or number of attack-patterns with the same prefix of a specific *prefixLength*.

$$Variability = \frac{1}{N} \sum (\text{prefixLength} * \text{numOfSeq}) \quad (4)$$

In the above  $N$  is the total number of unique sequences



(a)



(b)

**Figure 4.** (a) Average damage reduction vs error (ftp-write attack). (b) Average damage reduction vs error (eject attack).

of attack-patterns. A high value of variability indicates a high number of sequences with unique prefixes. Clearly, in this case the correct attack pattern can be recognized early which would allow less damage to occur and therefore, would correspond to the higher damage reduction.

Figures 3(b) and 4 present a damage reduction against two types of error:

- *Early response error*: Low threshold may force pre-emption long before the distinguishing aspects of different attack-patterns are seen (prefix of multiple patterns matches the sequence being monitored). As a result, monitored sequence may eventually end up in one attack-pattern while the response, selected and deployed pre-emptively, may correspond to some other attack. We will focus primarily on these type of errors; however it must be noted that any pre-emption can lead to such problems – the main challenge is to allow flexibility and capability to fine tune thresholds to vary pre-emption depending on prior results and security requirements.
- *Never-seen-before sequence*: when monitored sequence represents a novel pattern, although matches a prefix of some existing anomalous sequence. We claim that intrusion detection system is responsible for detecting new attacks and their addition to the attack-pattern repository. Responses can be attached only after the attacks are identified by the IDS.

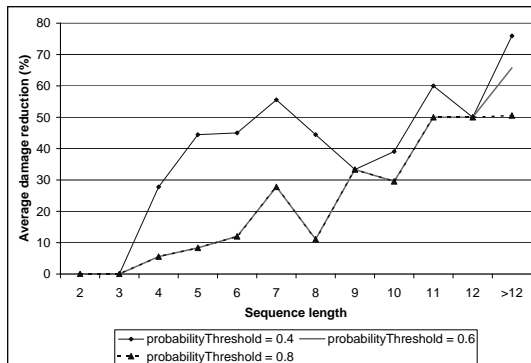
As expected, the results show that the error decreases with the increase of probability threshold value. With the delay of response selection process increases a confidence in some attack and consequently increases the accuracy of prediction of anomalous sequence.

Figure 5 presents a damage reduction from the perspective of average sequence length on the example of the *ftp-write* attack. Generally, damage reduction increases with the increase in sequence length. We can also note that this increase is more gradual for higher values of probability threshold (0.6, 0.8) and becomes very rapid for probability threshold = 0.4. This again confirms the observation that low probability threshold gives the highest amount of damage reduction. There are two sudden changes in graph patterns: one at sequence of size 8 and another at sequence of size 9. They represent two data extremes. Sequences of length 8 are almost identical with slight differences (low variability) at the end. Therefore, to reach specified probability threshold system waits until almost the end of these patterns, resulting in low damage reduction. On the other hand, a sequence of size 9 has a very unique pattern, allowing system to recognize it early with little influence of probability threshold value.

For the above experiments, it can be inferred that the systems focused on the significant damage reduction with the ability to tolerate higher error level can set up a lower probability threshold while systems more oriented on the accuracy of the deployed response should lean towards probability threshold value that is closer to 1.0.

## 5 Conclusion

In this paper we have presented a novel approach to cost-sensitive modeling of the response for pattern-based intrusion detection systems. The proposed model allows preemptive deployment of the response actions and their timely adjustment to the changing environ-



**Figure 5.** Average damage reduction vs sequence length (ftp-write attack)

ment. Via simulations we showed the dependency among probability threshold, damage reduction and error values. Practically, this allows systems with different security objectives to select an optimal combination of damage reduction and tolerable error.

In our future work, we intend to investigate a possibility of incorporating into our model a hierarchy of responses that will minimize the early response error. Automatic mapping of this hierarchy to the attack sequences will be another avenue of future research. In addition, we also plan to focus on real-time implementation of our algorithm which might give a deeper insight into the system advantages.

## References

- [1] I. Balepin, S. Maltsev, J. Rowe, and K. Levitt. Using specification-based intrusion detection for automated response. In *Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection*, 2003.
- [2] T. Bowen, D. Chee, M. Segal, R. Sekar, T. Shanbhag, and P. Uppuluri. Building survivable systems: An integrated approach based on intrusion detection and damage containment. In *Proceedings of the IEEE DARPA Information Survivability Conference and Exposition*, 2000.
- [3] C. Carver, J. M. Hill, and J. R. Surdu. A methodology for using intelligent agents to provide automated intrusion response. In *Proceedings of the IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop*, pages 110–116, 2000.
- [4] R. G. Coyle. *Decision Analysis*. The Camelot Press Ltd., London, GB, 1972.
- [5] B. Foo, Y.-S. Wu, Y.-C. Mao, S. Bagchi, and E. H. Spafford. ADEPTS: Adaptive intrusion response using attack graphs in an e-commerce environment. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 508–517, 2005.
- [6] J. B. Grizzard, S. Krasser, H. L. Owen, E. R. Dodson, and G. J. Conti. Towards an approach for automatically repairing compromised network systems. In *Proceedings of the 3rd IEEE International Symposium on Network Computing and Applications*, pages 389–392, August 2004.
- [7] S. Kumar and E. H. Spafford. A pattern matching model for misuse intrusion detection. pages 11–21, 1994.
- [8] W. Lee, W. Fan, M. Millerand, S. Stolfo, and E. Zadok. Toward cost-sensitive modeling for intrusion detection and response. In *Journal of Computer Security*, volume 10, pages 5–22, 2000.
- [9] M. E. Locasto, K. Wang, A. D. Keromytis, and S. J. Stolfo. FLIPS: Hybrid adaptive intrusion prevention. In *Proceedings of the 8th International Symposium on Recent Advances in Intrusion Detection*, pages 82–101, 2005.
- [10] MIT. Intrusion detection attacks database. Available from "http://www.ll.mit.edu/IST/ideval/docs/1999/attackDB.html".
- [11] T. R. Peltier. *Information Security Risk Analysis*. Auerbach Publications, 2001.
- [12] P. Porras and P. Neumann. EMERALD: event monitoring enabling responses to anomalous live disturbances. In *Proceedings of the National Information Systems Security Conference*, 1997.
- [13] F. Qin, J. Tucek, J. Sundaresan, and Y. Zhou. Rx: treating bugs as allergies—a safe method to survive software failures. In *Proceedings of the 20th ACM symposium on Operating systems principles*, pages 235–248, 2005.
- [14] D. Schnackenberg, K. Djahandari, and D. Sterne. Infrastructure for intrusion detection and response. In *Proceedings of the IEEE DARPA Information Survivability Conference and Exposition*, 2000.
- [15] R. Sekar and P. Uppuluri. Synthesizing fast intrusion prevention/detection systems from high-level specifications. In *Proceedings 8th Usenix Security Symposium*, 1999.
- [16] S. Sidiroglou, M. E. Locasto, S. W. Boyd, and A. D. Keromytis. Building a reactive immune system for software services. In *Proceedings of the USENIX Annual Technical Conference*, 2005.
- [17] A. Somayaji and S. Forrest. Automated response using system-call delay. In *Proceedings of the 9th USENIX Security Symposium*, 2000.
- [18] N. Stakhanova, S. Basu, R. Lutz, and J. Wong. Automated caching of behavioral patterns for efficient run-time monitoring. In *Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing*, 2006.
- [19] N. Stakhanova, S. Basu, and J. Wong. A taxonomy of intrusion response systems. *International Journal of Information and Computer Security*, 1(1/2):169–184, 2007.
- [20] T. Toth and C. Kruegel. Evaluating the impact of automated intrusion response mechanisms. In *Proceedings of the 18th Annual Computer Security Applications Conference*, 2002.
- [21] P. Uppuluri and R. Sekar. Experiences with specification-based intrusion detection. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection*, 2000.
- [22] G. White, E. Fisch, and U. Pooch. Cooperating security managers: A peer-based intrusion detection system. In *IEEE Network*, volume 10, pages 20–23, 1996.