Managing Business Health in the Presence of Malicious Attacks

Saman A. Zonouz†, Aashish Sharma†, HariGovind V. Ramasamy‡, Zbigniew T. Kalbarczyk†,
Birgit Pfitzmann‡, Kevin McAuliffe‡, Ravishankar K. Iyer†, William H. Sanders†, and Eric Cope‡
University of Illinois at Urbana-Champaign†, IBM Research ‡

Abstract

Business metrics play a critical role in determining the best system-level configuration to achieve an organizational business-level goal. We present a framework for reasoning about business-level implications of malicious attacks affecting information technology (IT) systems that underlie various business processes. Through an exemplar web-based retail company scenario, we demonstrate how to quantify both the relative value of the individual business processes, and the relative cost to the business caused by breach of key security properties. The framework allows for mapping business-level metrics to IT system-level metrics, and uses a combination of those metrics to recommend optimal response actions and to guide recovery from security attacks. We validate the framework against three high-impact attack classes common in such web-based retail company situations.

I. Introduction

As Information Technology (IT) infrastructures become increasingly complex and inter-dependent, there is a strong need to handle the infrastructure components not as individual silos, but as entities that need to be orchestrated in a coordinated manner that provides the most value for the business. In an environment of constrained IT budgets, enterprises are increasingly interested in tools and technologies for IT infrastructure optimization that factor in business value as the primary driver. Recent works have explored the linkage between IT system-level optimization actions and business value in the context of job scheduling, resource allocation, and performance, e.g., [3]. However, they do not address the provision of dependability in the presence of malicious attacks at the IT infrastructure level that simultaneously optimizes for the overall business value.

Optimizing business value in the presence of IT system-level attacks is a very complex problem for three reasons. First, it requires understanding of how the various business processes are mapped to the underlying IT resources. Until recently, this required extensive design documentation, e.g., design specifications. Such documentation is not only rare in practice but, even if present, also difficult to keep up-to-date. With the advent of automated discovery tools for fine-grained application dependencies, like Galapagos [3], [5], the ability to perform this mapping for large-scale IT infrastructures is only now starting to become a reality. Second, it requires understanding and quantification of the impact of violation of dependability and security (i.e., business health) supported by the IT system components underlying these business processes. Third, it is inherently more difficult to optimize the business value under security attacks than to achieve other system goals, such as performance and availability of a server, because a single security attack such as one resulting in a compromised credential may have diverse implications on the business and may affect multiple business processes.

In this paper[1], we describe a framework that enables reasoning about the business-level implications of attacks affecting the underlying IT systems in real enterprises. We demonstrate how to quantify system dependability/security at the business level, namely in terms of the value of the individual business processes, and the cost to the business caused by the breach of key system properties. The framework leverages an existing cost-sensitive automated intrusion response engine called the *recovery and response engine* (RRE) [8], and extends it to incorporate both IT system-level and business-level metrics. RRE uses the combined metrics to recommend optimal response actions to security attacks.

To illustrate the usefulness of our framework, we describe and model an exemplar web-based enterprise consisting of multiple business functions with varying levels of value and importance to the overall enterprise. We chose three main IT system-level attack classes from a pool of 150 security incidents observed in the National Center for Supercomputing Applications (NCSA) at the University of Illinois over the last five years [7], and implemented an incident-replay engine (IRE) to simulate their occurrence in the enterprise. We model the interaction between the attacker and the enterprise as a multi-step, sequential, hierarchical, non-zero-sum, two-player stochastic game between the IRE and RRE respectively. Our experiments demonstrate the superiority of our attack response approach over static intrusion response selection techniques

[1]The interested reader is referred to our complete technical report which is available at https://netfiles.uiuc.edu/saliari2/www/Tech-Report.pdf

and over techniques that do not consider business value.

II. Business Health Management Framework

RRE Model. We briefly describe the response and recovery engine (RRE) introduced in [8]. RRE models the security battle between itself and the attacker as a multi-step, sequential, two-player stochastic game. RRE uses competitive Markov decision processes (CMDPs) to model two controllers with conflicting interests, i.e., itself and attacker. In obtaining the solution for the CMDP (namely, the optimal response action) RRE tries to optimize the overall cost. RRE employs an extended attack tree model, called the *attack-response tree* (ART), to correlate IDS reports and formulate effects of response actions. More specifically, ARTs formally define "insecure system state," i.e. ART's root node, using logical `AND` and `OR` gates [8]. Nodes (consequences) in ART are recursively decomposed into more concrete (deeper) nodes in the tree leading to a set of leaf nodes (most concrete consequences) that are detectable by IDS alerts. Therefore, each leaf node in ART is mapped to its corresponding subset of IDS alerts. Each node is represented by a binary value depending on whether its corresponding consequence has already occurred in the system. If any of the IDS alerts mapped to a leaf node are received by RRE, the bit in that leaf node is set to `1`, which propagates bottom-up in ART to calculate the root node's value, i.e., whether the system is secure or not. Furthermore, some of the consequence nodes are tagged by a response action that, if taken, resets all leaf nodes in its underlying subtree to `0`.

Application of RRE to Business Health. In this section, we describe how a given IT system-level RRE model is augmented to include business-level considerations. Using business-level metrics in conjunction with standard IT system-level metrics allows RRE to prioritize actions that would result in optimizing business health, e.g., by keeping alive the most critical business processes and minimizing the user-visible service interruption. From an IT system perspective, the integrity of two database files may be equally important. But if one contains client credit card information while the other contains users' product ratings, they represent different business criticalities. Factoring that information as business-level metrics into the RRE helps prioritize the recovery of the former over the latter. Towards that end, we extend the basic RRE to consider both IT system and business aspects of the system when determining responses to malicious attacks.

As in our previous work that considered only IT-level metrics [8], we formulate the cost function of RRE using dynamic programming. The equation in Figure 1 formulates the cost function in RRE updated to include business-level metrics. Here, S is the state space; $V(.)$ is the value function used to solve for the optimal response action; T is the state transition probability function. The action set is partitioned into two subsets: R represents the set of response and recovery actions, and A represents the set of adversarial actions. `Dmg` or the damage function represents immediate business-level cost for each state transition $s \rightarrow s'$, which purely depends on the subset of consequences caused by the attacker as the result of the transition. `Cst` denotes the cost of taking a particular response action.

We now describe how the response action recommended by the RRE is the optimal one from a business perspective. The equation in Figure 1 recursively calculates the optimal cost function for each state s. The value function $V(s)$ is first initialized to zero for all $s \in S$; then, the value iteration algorithm [8] is employed to evaluate the correct value of the V function. The optimization involves two stages: the move of the attacker $a \in A$, and the move of the responder $r \in R$. At each step of the game s, RRE takes an action r making a transition to state s' from which the attacker takes next action $a \in A$ and makes a transition to state s''. The optimal response action is picked by minimizing over the response actions ($\min_{r \in R}$) the maximum possible damage caused by the attacker in next step (by maximizing over the adversarial actions in state s' using $\max_{a \in A}$). The $\lambda \in [0, 1]$ is the a fixed discount factor, e.g., $\lambda = 0.9$, to put more weight on present actions compared to the actions taken in future [8]. For a given business, the damage function `Dmg` depends on the primary service type provided by the business. Studies by others have attempted to provide estimates for average cost of downtimes and data breaches for different industry classes [1], [2], [4].

We now show how the numbers provided by studies such as [1], [2], [4] may be used to provide values for variables in the equation in Figure 1. Suppose that the current state is s, and that there are three possible response actions $R = \{r_1, r_2, r_3\}$ from which the optimal action should be selected. Each response action r is associated with a positive cost $\texttt{Cst}(r)$. Suppose that the state s represents an online shopping web server being down, and response action $r_2 =$ `Restart the Web Server` (resulting in transition to state s_2) fixes that problem in 30 seconds (or 0.0083 hours). If the restart can be effected remotely by automatic means (involving no operator time), then we have $\texttt{Cst}(r_2) = 0$ and $\texttt{Dmg}(s, s_2) = 0.0083 \times \$90000 = \$750$, where \$90000 is the cost per hour downtime for a catalog sales center type of business (from [2]). Similarly, for an attacker action $a_1 =$ `Compromise database` on a database that stores 50 customers' credit card records, resulting in transition from state s' to state s'_1, we will have $\texttt{Dmg}(s', s'_1) = 50 \times \$305 = \$15250$ where \$305 is the cost per breached credit card record (obtained from [1]).

We also modify the ARTs in the basic RRE [8] to account for business-level metrics. In the basic RRE, each business process in the enterprise would have its own set of ARTs corresponding to the IT-level components underlying the business process; we call these *IT-system level ARTs*. Each IT-system level ART would have been treated independently by the RRE to get its own corresponding optimal response action. We augment that model by introducing a *business-level ART*, which estimates the overall business health of the enterprise and is composed of *IT-system level ARTs* as follows: each leaf node in

$$V(s) = \min_{r \in R} \Big\{ \sum_{s' \in S} T(s,r,s').\{\texttt{Cst}(r) + \texttt{Dmg}(s,s') + \sqrt{\lambda}.\max_{a \in A}[\sum_{s'' \in S} T(s',a,s'').\big(\texttt{Dmg}(s',s'') + \sqrt{\lambda}.V(s'')\big)]\} \Big\}$$

Figure 1. The Dynamic Programming Equation to Solve for the Optimal Response Action

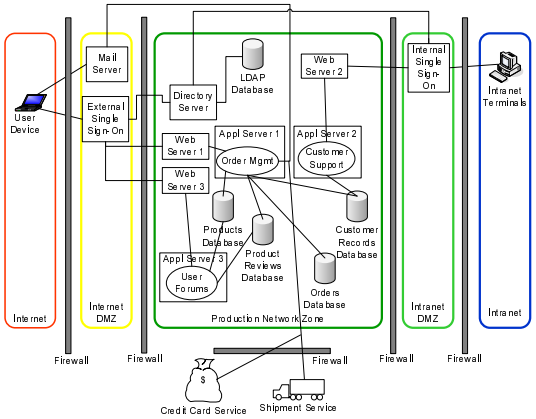


Figure 2. Enterprise IT Infrastructure

the business-level ART corresponds to a root node of an IT-level ART. The value of a root node of an IT-level ART indicates how likely it is that that component is currently secure. More specifically, it indicates violation of a security property in a corresponding IT system-level component of the enterprise. These values are later used to set the values of leaf nodes in the business-level ART. The overall business-level health of the enterprise is represented by root node of the business-level ART, and its value is calculated according to a bottom-up logical propagation of the values in its leaf nodes.

We update the response action set in the ARTs to include business- as well as IT system-level responses. For instance, business-level responses for a server intrusion may include hiring a red team, posting a "sorry, our web ordering is down" notice on the server, and sending coupons to clients. Additionally, the cost function in RRE is updated to address not only IT system-level responses but also business-level responses that the core response engine can choose from; in other words, responses tagged to business-level consequence nodes (e.g., "order management business process is down.") in ARTs also have associated costs assigned to them.

III. Case Study

To motivate and validate our business-value-driven intrusion response framework, we consider a typical web-based retail company architecture, called "Widgets R Us." In this section, we describe three of the company's business processes, its IT infrastructure, and how its business processes are mapped to its IT infrastructure. We also describe various attack scenarios.

Business Processes. We consider three business processes for "Widgets R Us"[2]: order management, customer support, and customer forum. The order management process is an abstraction of the typical high-level activities required to support a web-based retail transaction. The process includes activities performed by external service companies: a credit card verification service and a shipping service. For simplicity, the customer support process is limited to an abstraction of phone-based customer support. The process activities include answering general product questions and creating/updating customer account information. The customer forum process represents activities enabling customers to view, update, and discuss product reviews. When a customer browses the product catalog during the order management process, the order in which the products are presented to the customer is based, in part, on the user reviews.

These three retail-based business processes vary in the business value they provide to the company and, therefore, will have different security priorities associated with them. The order management process is the most critical. The company can incur significant costs associated with stolen information, for example credit card information [1]. The confidentiality, integrity, and availability of all the entities associated with the process are vital to the company's reputation and income; disruptions to the process have direct revenue impact. The customer support process is important, but less critical from a direct revenue impact standpoint than order management. Customer support is, however, important from a reputation standpoint. Since customer information is accessed and manipulated during the process, the availability and integrity of this information are areas of concern. The customer forum process is helpful to potential customers, but is the least critical of the three processes from a business value standpoint. Since product reviews affect product rankings, maintaining the integrity of reviews is a concern.

IT Infrastructure. Illustrated in Figure 2 are the various artifacts of the "Widgets R Us" IT system architecture. It is a canonical three-tier architecture, partitioned into multiple network zones for defense-in-depth. The production systems reside in a production network zone, separated from the Internet by a demilitarized zone. The production network zone is also separated from the company intranet. This separation restricts access to the production system to only authorized, internal users; for example, senior administrators that perform application server updates and maintenance.

There are two web servers connected to an external single sign-on (SSO) server: one supports the web content for the order management process, and the other supports the customer forum process. A third web server, accessible only from the company intranet, is connected to an internal SSO server. This server is used by the customer support process. Both of the SSO servers perform the roles of authentication proxy and reverse HTTP proxy. Internal and

[2]See https://netfiles.uiuc.edu/saliari2/www/Tech-Report.pdf

external web requests are first directed to their respective SSO servers. The SSO server examines the URL in the user's web request and determines to which of the web servers the request should be forwarded.

The three business processes are executed on separate application servers. Customer records, customer orders, product records, and product reviews are stored in separate databases with each database running on a dedicated server. An LDAP user registry and authorization database stores information about users (both internal and external), groups, and roles and authorizations (for example, which user entity can perform what operation on which IT system component). A directory server is connected to the LDAP database. The SSO servers interact with the directory server to check a requested URL against LDAP and authenticating the user as required. Lastly, there is a mail server that receives, routes, and delivers email.

Business-to-IT Mapping. Identifying what IT system components are needed to support each business process activity is a complex undertaking. Previous work (involving some authors of this paper) [3] describes how such a mapping can be performed in an enterprise setting. The first step is the identification of the IT entry points for each business activity. IT entry points are system components (for example, a server, an application or middleware) by which a human, external contact, or machinery access the IT infrastructure. Typically, these points correspond to the inputs and outputs of the IT system. For example, the web server in Figure 2 is the IT entry point for the product catalog browsing activity of the order management process. The second step is to identify the dependencies that each IT entry point has on other IT system components. Automated discovery systems such as Galapagos [5] can be used to discover fine-grained dependency information between applications, middleware, and data. In the case of the catalog browsing activity, such a discovery system will indicate that the web server forwards requests to an application server, which in turn accesses data from a database server. Thus, the product catalog browsing activity depends on three IT components.

Using the aforementioned mapping approach, the three business processes are mapped to the IT infrastructure in Figure 2 as follows: The order management process is mapped to Web Server 1, Application Server 1, Orders Database, Customer Records Database, and Product Reviews Database; the customer support process is mapped to Web Server 2, Application Server 2, and Customer Records Database; and the user forum process is mapped to Web Server 3, Application Server 3, Products Reviews Database, and Products Database.

Threat Model. We evaluate the effectiveness of our framework in the context of three attack classes: (1) user/admin credential compromise, (2) network sniffing, and (3) file system compromise. All three attack classes are prevalent, as evident in our incident analysis of IDS logs collected over the last five years at NCSA [7]. These three attack classes cover a broad range in parameters such as likelihood of their success, their potential disruption to the IT systems, and the cost to the company. Additionally,

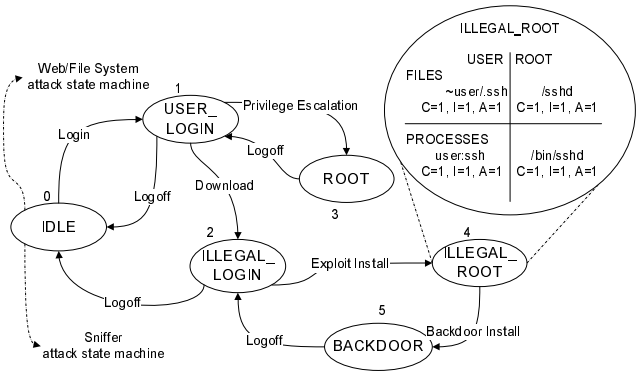


Figure 3. Credential Compromise Attack

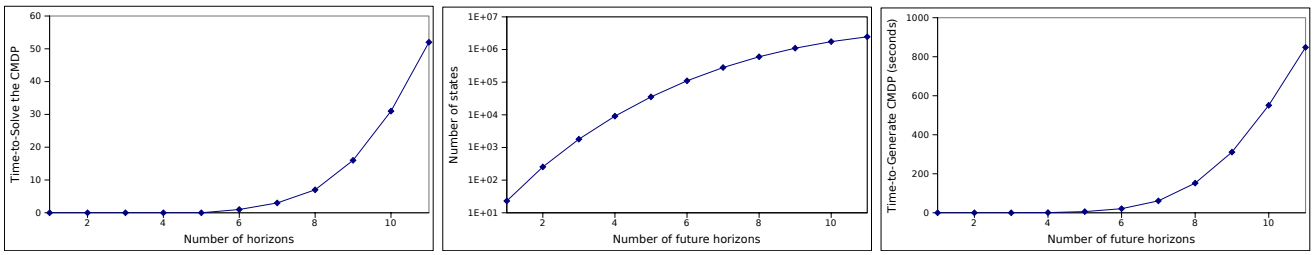
each of these attack classes could serve as a launch pad for several other attacks.

Credential Compromise: Attacks seeking user credentials are common. Supplemented with easily available local root escalation exploits (e.g., CVE-2009-2692), stolen user credentials can help attackers obtain total control of the systems inside the network. *Network Sniffing:* Once inside an enterprise network, the attacker can install a network sniffer to help gather passwords for instant messengers, POP, IMAP, VNC, and similar services. Armed with such sensitive information, the attacker can carry out even more damaging attacks, e,g., session hijacking on HTTP traffic and gaining access to employee emails. *File System Attacks:* Certain attacks on web-based ordering systems target the file system from which the web server is distributing web pages. For example, a vulnerability in the shared file system (e.g., NFS) may allow a remote attacker to read/write contents in a directory with user or root privileges.Such an exploit would allow the intruder to run a fake web site on a genuine corporate server and lure customers into providing sensitive data.

IV. Evaluation

We evaluate how considering business-level security metrics in conjunction with IT system-level metrics affects decision quality in terms of the cost of recovery after an attack. RRE is currently implemented for both Windows (C#.Net) and Linux (C/C++). All experiments were conducted on a system with a 2.2 GHz AMD Athlon 64 Processor 3700+ with 1 MB of cache, 2GB of memory, and the Ubuntu OS (Linux 2.6.24-21).

Incident Replay Engine (IRE). We constructed an incident-replay engine (IRE) based on attack patterns and monitor alerts obtained from the analysis of 150 attack incidents observed at NCSA (UIUC) over five years [7]. In essence, IRE mimics the world outside the RRE. At the heart of the IRE are sets of finite state machines. Each finite state machine (FSM) is an attack model for incidents belonging to the same attack class. For example, Figure 3 shows the state machine model for incidents grouped under credential compromise, which accounts for 26% of all analyzed incidents. Each node in the FSM depicts an IT



(a) Solution Time for the CMDP (b) State-Space Size (c) Time to Generate the CMDP

Figure 4. Response and Recovery Engine Performance

system-level state. The arcs (transitions) are the actions taken by a genuine user or an attacker, or alerts output by the monitoring system. Some transitions can be performed in multiple ways, e.g., an attacker can use different exploits to achieve his/her goal. A specific alert is associated with each transition. While replaying incidents, IRE also accounts for deficiencies of the real-world monitoring infrastructure by generating false positives. Building the FSM entails the following steps: (1) identification of all alerts associated with a given category of incidents; (2) identification of distinct system/application states traversed by the attacker in penetrating the network; and (3) assigning conditions to transitions between the system states identified in step 2.

IRE emulates the actions of both legitimate users and attackers using a state-based Markov model. It generates actions stochastically based on an exponential distribution as in the TPC-W benchmark [6]. Each action is associated with a set of pre-conditions and post-conditions; pre-conditions need to be met before the action can be taken, and post-conditions represent the action's effects on the system state variables. For instance, if IRE generates the attacker action "remove the `/etc/passwd`" (which requires root access), and the attacker only has user access privileges, then IRE simply ignores the action. On the other hand, if the attacker has already obtained root access, then the IRE updates the system state to reflect the removal of the file. At the end of each step, IRE doubling as a detection system, also sends reports to the RRE regarding attack incidents in the system. The reports that IRE sends to RRE include the confidentiality, integrity, and availability (CIA) binary values for files and processes of interest (e.g., the `/etc/passwd` file has been accessed and the `sshd` process terminated). Upon receiving reports from IRE, the response engine (RRE) concurrently (in a separate process) decides upon optimal response actions and sends the selected action back to IRE. Finally, IRE updates the system state to reflect the completion of the response action.

RRE Performance Analysis The business cost of a successful attack is directly affected by the complete recovery time which consists of two factors: the time to select the response action, and the time to carry out the action. RRE uses the ART for the case study network with 42 leaves, which is fed into RRE in XML format. The ART is automatically converted to its corresponding CMDP, that is later solved for response actions. ART-to-CMDP conversion of the ART results in a CMDP with 2^{42} states, which is computationally intractable; therefore, RRE makes use of an approximation algorithm, i.e., *envelope* [8], to limit the lookahead steps (horizons) while solving the CMDP.

As shown in Figure 4(a), RRE can solve for response action within 10 seconds if the number of horizons to consider is kept below 8, and it takes about 1 minute if 11 horizons are considered. Clearly, the more horizons RRE considers, the closer the selected response action will be to the optimal one, but the more time it takes for the decision. Figure 4(b) shows the number of states given the number of future horizons RRE takes into account. For instance, considering only one future action sequence, the generated CMDP will have 43 states. The size of the state space exceeds one million when the number of horizons is set to 11. RRE's most time-consuming task is the conversion of the business-level ART into its corresponding CMDP model. Figure 4(c) shows that it takes about 10 minutes for RRE to accomplish the ART-to-CMDP conversion when 10 future horizons are considered. However, it is important to mention that the ART-to-CMDP conversion step could be a one-time effort and done only once off-line, before the whole framework starts to operate.

Comparison with Static Response Mechanisms. We evaluated RRE against response systems that statically choose responses from a lookup table containing (IDS-alert, response) mappings. There are a total of $2^{|\mathcal{L}|} = 2^{42}$ initial system states for the interaction; we investigate how RRE and the static engines perform, in terms of recovery cost (i.e, the total time to recover in seconds), when $1 \leq i \leq 42$ of the IDS alerts are received in the starting state. As shown in Figure 5, RRE takes advantage of its game-theoretic cost optimization engine to recover the system with lower total cost. It is important to note that the recovery costs for the two engines are closer to each other when there are fewer IDS alerts received, and hence fewer leaf nodes fired in RRE; however, RRE is more helpful than static engines when larger numbers of IDS alerts from different parts of the system are received. The reason is that RRE solves CMDP to get the optimal action while the static engine simply looks up its manually prefilled table and sequentially takes response actions corresponding to each received alert.

RRE with and without Business-level ART. We evaluated the business-level impact of using the extended RRE (that includes the business-level ART), the basic RRE (presented in [8]), and a static intrusion response system

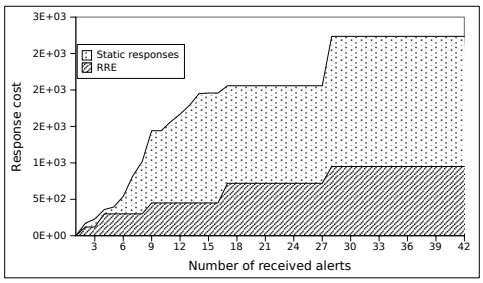


Figure 5. RRE vs. Static Intrusion Response

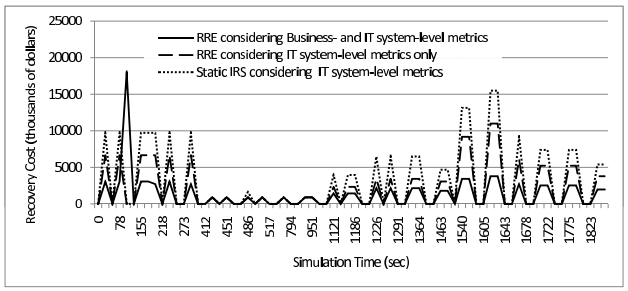


Figure 6. Recovery Cost Comparison

for responding to attacks. We performed the evaluation in the context of the Widgets R Us enterprise example. Consider a situation in the Widgets R Us example where two servers, one mapped to the order management business process and the other mapped to the user forums business process, are simultaneously compromised by attackers. By exploiting the business-level metrics, the extended RRE could prioritize the compromised servers with respect to their criticality to overall business health and deal with them in that order, e.g., handling the server that deals with the order management business process first. The extended RRE does this prioritization by making use of the costs of the consequences nodes in ART. That would not be possible in the case of the basic RRE, which considers only IT system-level metrics; with basic RRE, there is no way to distinguish the criticality of the various applications/servers to the overall business health of the whole infrastructure.

Figure 6 illustrates the discounted recovery cost during a 30-minute system-attacker interaction in which IRE emulated the attack incidents and the IT infrastructure of Widgets R Us and continuously sent IDS alerts out to RRE, which was running in a separate process. As explained before, the cost to recover from an attack depends both on the time it takes for the response engine to select the response action, and the time it takes to effect that response action. To permit fair comparison, the same cost function was used for both the IT-level ART and business-level ART models; similarly, the same IRE model was used for both. Given the received alerts, RRE started the game-theoretic decision engine to decide upon the optimal response action r using the equation in Figure 1. We assigned the cost per hour downtime for the Widgets R Us enterprise to be $\$180K$, which is the cost presented in [2] for Amazon.com.

RRE sends the chosen optimal action (which could be NO-OP) to the IRE, whose agents receive and execute the action(s) and update the state of the infrastructure. It is possible for the clients (or attackers) emulated by IRE to take their next normal (or adversarial) action(s) while RRE was still deciding upon what response action to take; in those situations, RRE's recommended response action was received by IRE when the infrastructure was in a state different from the one for which the response action was chosen. Therefore, the received response action were not always optimal, i.e., RRE payed a higher cost to recover the system. In our experiments, 3% of RRE's actions, on average, were received by IRE after the next state transition had already happened. As shown in Figure 6, basic RRE had a lower recovery cost than the static intrusion response system. The recovery cost was even lower with the extended RRE. In summary, the results show that adaptive intrusion response systems can benefit by factoring in business-level metrics along with standard IT system-level metrics.

V. Conclusion

In this paper, we presented a framework for managing business health in the presence of malicious attacks. The framework addresses situations in system administration that involve multi-objective decision-making while taking into account both business- and IT system-level metrics. We extended RRE's ART formalism to enable the assessment of overall business health (i.e., dependability and security). We demonstrated the use of the framework in the context of an online enterprise against three common attack classes. Our work highlights the importance of considering business-level metrics in developing automated intrusion response systems. Without such consideration, there is the danger that actions recommended by an intrusion response system may not be relevant from a business perspective even if they are optimal at the IT system-level.

References

[1] *PCI Compliance and the cost of a credit card breach.* http://www.braintreepaymentsolutions.com/blog/pci-compliance-and-the-cost-of-a-credit-card-breach.

[2] 2009 Annual Study: Cost of Data Breach (Understanding Financial Impact, Customer Turnover, and Preventive Solutions), 2010. Ponemon Institute.

[3] N. Joukov, B. Pfitzmann, H. V. Ramasamy, N. G. Vogl, M. V. Devarakonda, and T. Ager. ITBVM: IT Business Value Modeler. In *Proc. 2009 IEEE International Conference on Services Computing*, pages 128–135, 2009.

[4] R. W. Kembel. *Fibre Channel A Comprehensive Introduction.* Northwest Learning Associates, inc., 2009.

[5] K. Magoutis, M. Devarakonda, N. Joukov, and N. Vogl. Galapagos: Model-driven discovery of end-to-end application-storage relationships in distributed systems. *IBM J. Research*, 52:367–378, 2008.

[6] D. Menasce. TPCW-Benchmark for E-Commerce, 2002.

[7] A. Sharma, Z. Kalbarczyk, R. Iyer, and J. Barlow. Analysis of credentials stealing attacks in an open networked environment. In *NSS*, 2010.

[8] S. A. Zonouz, H. Khurana, W. H. Sanders, and T. M. Yardley. RRE: A game-theoretic intrusion response and recovery engine. In *DSN*, pages 439–448, 2009.