IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

IDS Based: Response and Recovery Engine

Jivan Sunklod[1], Rahul Jadhav[2], Gangadhar Survyanshi[3], Digambar Gadhaling[4], Dr. Kishor Wagh[5]

Department of Computer Engineering GHRCEM, Savitribai Phule Pune University, Pune[1, 2,3,4,5]

Abstract: In today's life the internet user population are very increased that why the user face of fast-spreading intrusion. The intrusion detection possible not only detection algorithms, but also it required the special tool hence we create a new tool that is intrusion response and recovery in short RRE tool. In this paper, we propose a new product this product can do the automated response the intrusion this is called the Response and Recovery Engine (RRE). Our engine employs a user data transaction response strategy against adversaries modeled as opponents in a two-user stochastic transaction. Our software whose name is RRE involves attack-response trees to response the attacker and analyzes undesired security events and their countermeasures using Boolean logic to combine lower-level attack consequences. In addition, RRE database involve only the users data those who registered in the RRE. The product involves the intrusion detection system which detects the intrusion in Boolean form. RRE then correct option to take optimal response actions by solving a partially overt competitive Markov decision process that is automatically derived from attack-response trees. Experimental results show that RRE, using the Snort's alerts, the snorts alert can provide the security for networks for which assailment-replication trees have more than 500 nodes.

Keywords: Response and Recovery Engine (RRE), IP fragmentation, SMTP mass mailing, DoS attacks.

1. INTRODUCTION

To reduce impact of intrusion by detecting the type of attack on the system and provide optimal response. For a large network of computer user is increased then the system is deployed in an area, there are number of systems. The use of internet is in the order of increasing order of size in day to day life hence it is essential to provide the security of the effect of the security of the network is in great manner. IP fragmentation, SMTP mass mailing, DoS attacks, flood attacks, spoofing, buffer overflow are some of kind of the attacks that occur in the network. And one another serious issue in network it said to be network Intrusion. The systems are damaged from the intrusion. The need to reduce problem of protection maintenance of computer networks is one of the important thing. The main aim is to reduce this intrusion and take an correct action on the intrusion that save system damage and provide proper response to the intruders. Intrusion response usually is declared to be a manual process performed by network admin user who are notified by IDS alarms and detect to the intrusions. This manual response process inevitably take some time between notification and response, now this create the problem persistence again as the response can be easily exploited by the attacker. Genetic algorithm used for IDS were the most appropriate techniques for intrusion alerts but our aim to provide proper automated response was the main motivation. RRE was given the preference as it can be satisfy all the requirements, to be headlined we get the technique for automated response that provide reduction of intrusion response cost and intrusion response time.

In this paper, we present an automated low cost intrusion response system that system is called the Response and Recovery Engine (RRE) that designs the security battle between itself and from the attacker as a multi-step, sequential, hierarchical, non-zero-sum, two-user transaction. In each step of the transaction, RRE involved a new advanced attack tree like structure, called the attack-response tree (ART), and it received IDS alerts to evaluate various security properties of the system.

ARTs provide a leangle way to describe system security based on possible intrusion and response scenarios for the attacker and detect and response engine, respectively. More usefully ARTs are able RRE to consider inherent uncertainties in alarms received from IDSes (i.e., false positive and false negative rates) when counting the system's security and deciding on response actions. Then, the RRE automatically transformed the attack-response trees into partially observable competitive Markov decision processes that are solved to search the optimal response action opposite to the attacker, in the sense that the more discounted accumulative damage that the attacker can cause after in the transaction is minimized.

Using this game-theoretic approach, RRE deceptively adjusts its activities according to the attacker's possible in the future reactions, thus preventing the attacker from causing significant damage to the system by taking an intelligent action-chosen sequence of actions. To deal with security issues with different angularities, RRE's two-layer architecture consists of local engines, which reside in individual host computers, and the global engine, which stayed in the response and recovery server and decides on global response actions once the system is not able recover by the local engines. Furthermore, the hierarchical architecture improves scale ability, ease of design, and performance of RRE, so that it can provide security computing assets against attackers in large-scale computer networks.

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

2. RRE'S HIGH-LEVEL ARCHITECTURE

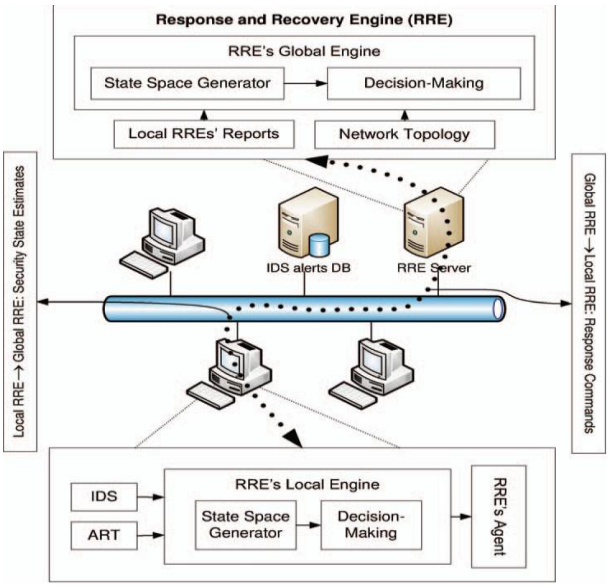

Figure 1: RRE's High-level Architecture

Before taking theoretical design and enforcement details, we constitutionalize a high-level architecture of RRE, as declared in Fig. 1. It has two types of decision-making engines Figure 1. High-level architecture of the RRE at two distinct layers, i.e., local and global. This is stratified structure of RRE's architecture (discussed later) makes it capable of handling very dominant IDS alerts, and selecting optimal response actions. More than, the two-user architecture improves its quantifibiality for large-scale computer networks, in which RRE is supposed to provide security of a large number of host computers from the malicious attackers. Finally, partition of high- and low-level security issues significantly simplifies the correct design of response engines. As the first layer, RRE's local engines are separated in host computers. Their main set of inputs involves of IDS alerts and attack-response trees. All IDS alerts are sent to and stored in the alert database (Fig. 1) to which every local engine subscribes to get notified when any of the alerts related to its host computer is received. Using the preceding local information, local engines compute local response actions and send them to RRE satisfier that are in charge of enforcing received commands and reporting back the recruitment status, i.e., whether the command was successfully take out. The internal architecture of engines includes two major components: one is the state space generator and the other is decision engine. Once inputs have been received, all possible cyber security states in which the host computer can be are generated. As discussed later in, the state space might be intractably large; therefore, RRE partially creates the state space so that the decision making unit can quickly decide on the optimal response consultation. The decision-making unit employs a pattern matching algorithm that models attacker-RRE interaction as a two user in which each user tries to maximize his or her overall benefit. This implies that, once a system is under attack, immediate grasping response decisions are unnecessarily the best choices, since they may not guarantee the minimum total accumulative cost involved in fully recovery from the attack. Although individual local engines attempt to provide security to their corresponding host computers, global network-level response actions required inputs from multiple host computers. Moreover, individual local engines may become malicious contain if they get compromised. To Handel to these problems, RRE's global engine, as its second layer, gets high-level information from all host computers in the network, orientate on optimal global response actions to take, and organize RRE agents to accomplish the actions by sending them pertinent response commands. In addition to local security calculate from host computers, network topology is also nourished into the global engine in the form of an ART graph, which shows what combinations of compromised host computers will change the security status of the whole network, and what global response actions are available to stop the attacks. The ARTs, in the global engine, depend upon the network's topology; therefore, they should be specifically designed by experts for each network. In opposition, the ARTs for local assets, once designed, are simply re-used. If the global engine is cooperating, the ability to implement global-level response actions is affected; however, the local engines are not affected. Moreover, global engines can be protected by employing equal to security measures, and if feasible, intrusion tolerant approaches.

3. LOCAL RESPONSE AND RECOVERY

Having given a high-level overview of how in hierarchical manner structured components in RRE interface with each other, we now present the theoretical form of design of these components in detail. Starting with the lowest-level component in RRE, we explain in details how local engines, residing in host computers, provide security to local computing assets using security-related data, i.e., IDS alerts, about them.

3.1. Attack-Response Tree

To forefend a local computing asset, its corresponding local engine first endeavors to deduce what security properties of the asset have been assailed, given a received set of alerts. Attack trees offer a convenient way to systematically categorize.

3.2. Stackelberg Game:

RRE vs. Assailant Reciprocal interaction between the adversary and replication engine in a computer system is genuinely a game in which each player endeavors to maximize his or her own benefit. The replication cull process in RRE is modeled as a sequential Stackelberg stochastic game in which RRE acts as the bellwether while the assailer is the adherent; however, in our illimitable-horizon game model, their roles may change without affecting the final solution to the quandary. Categorically,

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

the game is a finite set of security states S that cover all possible security conditions that the system could be in. The system is in one of the security states s at each time instant. RRE, the bellwether, culls and takes a replication action ms ∈ M admissible in s, which leads to a probabilistic security state transition to s.

3.3. Automatic Conversion:

ART-to-MDP Utilizing the ARTs, RRE's local engines automatically construct replication decision process models, where security states are defined as a binary vector whose variables are genuinely the set of satiated/unsatisfied (1/0) leaf consequence nodes in the ART under consideration. In other words, as a binary string, each security state vector represents the leaf node consequences that have already been set to 1 according to the received alerts from IDS systems. For instance, the security state space for an ART with n leaf nodes consists of 2n n-bit state vectors. For ARTs with an astronomically immense number of leaf nodes,

3.4. Agents

In the above mentioned security battle between RRE and the adversary, agents play a key role in accomplishing each step of the game. They are in charge of taking replication actions decided on by RRE engines. Authentically, having received commands from engines, agents endeavor to carry them out prosperously and report the result, whether they were prosperous or not, back to the engine. If the agent's report denotes that some replication action has been taken prosperously, the engines update their ART trees' corresponding variables, which are leaf node values in the subtree for the prosperously taken replication action node. Consequently, as explicated above, leaf node variables in ART trees are updated by two types of messages: IDS alerts and agents' reports. Otherwise, if the agent cannot respond prosperously (e.g., within a categorical duration), the second-best action is sent by the engine to carry out.

4. GLOBAL RESPONSE AND RECOVERY

Albeit host-predicated intrusion replication is taken into account by RRE's local engines utilizing local ARTs and the IDS rule-set for computing assets, e.g., the SQL server, maintenance of ecumenical network-level security requires information about underlying network topology and profound understanding about what different amalgamations of secure assets are compulsory to assure network security maintenance. In RRE, ecumenical network intrusion replication is resolved in the ecumenical server, where, just as in local engines, ARTs are utilized for correlating received information, and then maximin theory is applied to cull the optimal ecumenical replication action. Such a cull is not possible in local engines due to either their inhibited local information or their inability to manage cooperation among distributed RRE agents. In contrast to ARTs in local engines for computing assets that demand one-time design effort for each asset (as in IDS rule-sets), ecumenical ARTs in RRE's server for network security should be designed categorically for each individual network in which RRE is deployed, since these higher- ARTs depend on network topology, which implicitly affects (7) a network's ecumenical security state. In our current implementation, there is only one ecumenical, i.e., network-level, ART in RRE that must be designed by an expert. Generally, ecumenical ARTs in RRE's server have the same structure discussed in § 4.1, though some demystifying explications are needed regarding their root and leaf nodes. As discussed in § 4.4, local engines send their local security estimates, i.e., root node probabilities δg of their ARTs, toRRE's server. Indeed, local security estimates contribute to leaf nodes in the ecumenical ART in RRE's server. Furthermore, the top-event node of the ART in the ecumenical engine is labeled "network security infringement," and is defined and formulated according to the underlying nodes. In other words, network security is defined by the ecumenical ART in RRE's server utilizing its leaf nodes, which are themselves root nodes of local

ARTs in RRE's local engines. Ecumenical ART is employed for quantitative evaluation of a network's security state. The correlation and replication cull calculations are equipollent to in local ARTs (§ 4), except that for ecumenical ART, the rudimental leaf node l probability measures are computed as $\delta(l) = \delta g(l)$, where g(l) denotes the local ART corresponding to leaf node l of the ecumenical ART in RRE's server.

5. CASE STUDY

SCADA Networks In this section, we describe the replication cull process for a case study process control network for a puissance grid. We have culled supervisory control and data acquisition (SCADA) networks as our case study for two reasons. First, since they control physical assets, they require timely replication in the presence of attacks. Second, in contrast to general IT computer networks, they consist of computing assets with predefined categorical responsibilities and communication patterns; this simplifies the design process for comprehensive ARTs and IDSes with exhaustive alert sets. Fig. 3 shows a sample 3-bus power grid and its SCADA network, which is responsible for monitoring and controlling the underlying power system. There are a total of three engenderers, any one of which is able to provide the potency required by customers, i.e., load. To monitor the potency system, each bus is affixed to a sensor, i.e., a phasor quantification unit (PMU). The sensor sends voltage phasors (i.e., magnitudes and phase angles) of the bus and current phasors of transmission lines connected to that particular bus to SCADA. Moreover, to control power generation, having received sensory data, SCADA computes optimal generation set points for individual engenderers. As shown in Fig. 3, SCADA consists of different components, among which there are constrained communications. First of all, given sensory data, the state estimation server is responsible for estimating the state of

IJARCCE
ISSN (Online) 2278-1021
ISSN (Print) 2319 5940
International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

the whole power system. A database stores these states and other information that might be used later by administrators or customers through the web server. The human machine interface (HMI) and security constrained optimal power flow (SCOPF) compute Case study: SCADA networks. Attack - replication tree control commands utilizing those estimated states. As demonstrated, a sultry spare HMI is additionally active and connected as part of the network. a sample brief network-level ART for the process control network described above. The top event is opted to be "SCADA is compromised," and its children denote deficiencies in providing loads and report generation, which are two main goals of the supervisory network. For simplicity, leaf nodes here denote compromise of individual host systems, and are updated by local engines. As a case in point, G1, if set to 1, designates that the controller contrivance for the engenderer on bus 3 is compromised, and the upper replication node "restart" shows that a countermeasure for the compromised controller is to reinstall the control software and restart the contrivance. Details such as action costs, rates, and probabilities are not shown.

The cyber-security state space definition for the above attack-replication tree is shown in Fig. 5 as a binary vector, where each individual bit is set predicated on reports from local engines. For instance, the sample state vector in the figure be tokens that HMI and G1 are compromised, according to reports from their corresponding local engines, while other hosts are in their mundane operational mode. Given the assailment-replication tree and reports from local engines, RRE commences online construction of its accompanying partially overt competitive Markov decision process. Starting from the current state, i.e., s = (000100000010). A sample cyber-security state there are a total of 13 possible transitions, partitioned into two subsets: 1) replication actions Ar(s) = {restart(G1), switch(HMI),NOP}, and 2) adversarial actions Aa(s) regarding leaf consequence nodes. Here, we surmise that responsive actions Ar(s) require some manual assistance by a SCADA operator; hence, they cannot be accomplished simultaneously due to constrained human resources. The solution for this model by RRE gives switch (HMI) as the optimal replication action, since if restart (G1) or NOP was culled, the assailer could cause an abundance of damage to the system afterwards by compromising the SCOPF server (Fig. 4), leading to consummate failure of the control subsystem that would consequently affect how power loads were provided and conclusively result in the top event "SCADA compromised." In other words, as explicated earlier, the engine culls the replication action that minimizes the maximum damage that the assailant can cause later.

6. COMPUTATIONAL EFFICIENCY

Albeit the value iteration algorithm performs well in MDPs with several thousand states, RRE (like most state based modeling techniques) faces the state space explosion quandary when a sizably voluminous network that includes an astronomically immense number of assets is to be for fended utilizing numerous alerts sent by distributed IDS systems. This exponential magnification of the state space makes it infeasible to compute an optimal solution, i.e., replication action, in immensely colossal-scale applications. The quandary becomes even worse when POCMDP is employed to find an optimal solution. Consequently, RRE uses two state compaction techniques to deal with this quandary.

First, the most likely state (MLS) approximation technique [5] is utilized to convert POMDP to MDP, which is more tractable for authentic-time replication decision-making. To do so, we compute the most likely state utilizing $s* = \text{argmax}_s b(s)$, and define policy as $\pi(s) = \pi_{MDP}(s*)$, which is computed utilizing Bellman's optimality equations for the value function $V*(s) = \max_{ar \in Ar(s)} Y(V*, s, ar)$ and policy $\pi_{MDP}(s) = \arg\max_{ar \in Ar(s)} Y(V*, s, ar)$, in which Y(.) is as defined in (13). The value iteration algorithm[3] is employed to compute the value function, i.e., $V(s) \leftarrow \max_{a \in A(s)} Y(V, s, a)$. Utilizing MLS in RRE is quite plausible, since the probability of the most likely state is far more preponderant than the probability of other states; however, the derived MDP is not yet diminutive enough to deal with in authentic-time.

Furthermore, due to its astronomically immense state space, even off-line solution techniques are not utilizable, since most of them, e.g., value iteration, perform iterative updates over the entire state space. To focus computations on pertinent states, an online anytime algorithm1 called envelope is employed, making RRE capable of deciding authentic-time replications even in astronomically immense-scale computer networks. In brief, the envelope algorithm performs a finite 1An anytime algorithm can be interrupted at any point during execution to return an answer whose value, at least in certain classes of stochastic processes, ameliorates in prospect as a function of the computation time. look-ahead search on a subset of states reachable from a given current state, i.e., $s*$ (mentioned above).

This subset, called "envelope $E\pi$," initially contains only the current state and is progressively expanded. An approximate value function ˜ V is utilized to evaluate the fringe states, i.e., the set of states that are not in the envelope but may be reached in one step from some state in the envelope: $F\pi = \{s \in S - E\pi | \exists s \in E\pi, P(s, \pi(s), s) > 0\}$. (14) The envelope converges to the optimal policy [7], and its general scheme is as follows: 1) Initialization: Engender the initial envelope $E\pi = s*$. 2) While ($E\pi = S$) and (not deadline) do – Fringe expansion: Elongate the envelope $E\pi$. Some $s \in F\pi$ is culled, and its value is updated. – Forebears update phase: Engender an optimal policy π for the envelope. 3) Return π. Utilizing the envelope, RRE can solve immensely colossal MDPs very efficiently by engendering partial policy, defined only on the envelope, without evaluating the entire state space.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

7. EXPERIMENTAL EVALUATION

In this section, we investigate how the proposed Replication and Recuperation Engine performs in authenticity. We have implemented RRE on top of Snort 2.7 [22], which is an open-source signature-predicated IDS. The experiments were run on a computer system with a 2.2 GHz AMD Athlon 64 Processor 3700+ with 1 MB of cache, 2 GB of recollection, and the Ubuntu (Linux 2.6.24-21) operating system.

7.1. Scalability

To evaluate how RRE handles involutes networks consisting of astronomically immense numbers of host systems, we quantified the time required by RRE to compute the optimal replication action vs. sundry metrics. Fig. 6 shows the average timeto- replication over ten runs vs. the assailment-replication tree order, i.e., the maximum number of children for each node. Given a fine-tuned number, e.g., 500 in Fig. 6, of total nodes, the ART tree order determines the number of leaf nodes that contribute to the size of the state space in a Markovian decision model. For each tree order d, a balanced tree, in which each node has d children, is engendered; gates are assigned to be AND or OR with equal probability, i.e., 0.5.

In our experiments, the -optimality termination criterion in Bellman's equation and discounting factor are set to = 0.1 and $\gamma = 0.99$, respectively. Then, a decision process is constructed and solved, and the total time spent is recorded (Fig. 6). As expected, the figure shows that incrementing the ART order leads to rapid magnification of the required time-toresponse by the engine. In another scalability evaluation experiment, we quantified time-to-replication vs. the number of nodes in balanced ART trees of order 2. Fig. 7 shows average results on ten runs for two schemes.

First, given IDS alerts and the ART tree, the consummate decision model consisting of all states in the state space was constructed. As shown in Fig. 7(a), the replication engine can solve for optimal replication actions for ART trees with up to 45 nodes within about 2 minutes. Second, an online finite-look ahead Markovian decision model with an expansion limit of 2 steps was engendered and solved. Y(V, s, a) = _ s_∈ S P(s _|s, a).{r(s, a, s _) + √ γ.[min aa∈ Aa(s_) _ s__∈ S P(s __|s _ , aa). _ r(s _ , aa, s __) + √ γ.V (s __) _]} (13) As illustrated in Fig. 7(c), constrained expansion ameliorates a solution's convergence speed and increases the solvable ART size to trees with up to 500 nodes within 40 seconds.
By solving ART trees with about 900 nodes in a minute, RRE can forfend immensely colossal-scale computer networks. Third, to further amend RRE's scalability, we evaluated how expeditious a decision process is solved with an upper expansion limit of 2. Fig. 7(b) compares total instauration cost between the abovementioned two schemes for all possible starting scenarios, i.e., states (2|L| = 64).

8. CONCLUSION

A game-theoretic intrusion replication engine, called the Replication and Instauration Engine (RRE), was presented. We modeled the security maintenance of computer networks as a Stackelberg stochastic two-player game in which the assailant and replication engine endeavor to maximize their own benefits by taking optimal adversary and replication actions, respectively. Utilizing an elongated attack tree structure called the Attack-Replication Tree (ART), RRE explicitly takes into account inaccuracies associated with IDS alerts in estimating the security state of the system. Moreover, RRE explores the intentional malevolent attacker's next possible action space afore deciding upon the optimal replication action, so that it is ensured that the assailer cannot cause more preponderant damage than what RRE soothsays. Experiments show that RRE takes congruous countermeasure actions against perpetual attacks, and brings an insecure network to its mundane operational mode with the minimum possible cost.

ACKNOWLEDGMENTS

This material is predicated upon work fortified by the National Science Substratum under Grant No. CNS-0524695, as a component of the NSF/DOE/DHS Trustworthy Cyber Infrastructure for Power Center (http://tcip.iti.illinois.edu). We thank our colleagues, who provided insight and expertise that greatly assisted the research.
We thank our Mentor **Dr. Kishor Wagh** for guiding us for this project. We thank our Professor Sarita Patil for inspiring us to do this project and her comments on the manuscript Power Center (http://tcip.iti.illinois.edu).

REFERENCES

[1] A. Avizienis, J. Laprie, B. Randell, and C. Landwehr. Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. on Dep. and Sec. Comp., 1:11–33, 2004.
[2] I. Balepin, S. Maltsev, J. Rowe, and K. Levitt. Using specification-based intrusion detection for automated response. Proc. of the Int'l Symp. on Recent Advances in Intrusion Detection, pages 136–54, 2003.
[3] R. Bellman. Dynamic Programming. Princeton University Press, 1957; republished 2003.
[4] M. Bloem, T. Alpcan, and T. Basar. Intrusion response as a resource allocation problem. Proc. of Conf. on Decision and Control, pages 6283–8, 2006.
[5] A. Cassandra. Exact and Approximate Algorithms for Partially Observable Markov Decision Processes. PhD thesis: Brown University, 1998.
[6] F. Cohen. Simulating cyber attacks, defenses, and consequences. Journal of Comp. and Sec., 18:479–518, 1999.
[7] T. Dean, L. Kaelbling, J. Kirman, and A. Nicholson. Planning under time constraints in stochastic domains. Artificial Intelligence, 76:35–74, 1995.
[8] J. Filar and K. Vrieze. Competitive Markov Decision Processes. Springer-Verlag, 1997.
[9] B. Foo, M. Glause, G. Howard, Y. Wu, S. Bagchi, and E. Spafford. Information assurance: Dependability and Security in Networked Systems. Morgan Kaufmann, 2007.

IJARCCE

ISSN (Online) 2278-1021
ISSN (Print) 2319 5940

International Journal of Advanced Research in Computer and Communication Engineering
ISO 3297:2007 Certified
Vol. 5, Issue 11, November 2016

[10] B. Foo, Y. Wu, Y. Mao, S. Bagchi, and E. Spafford. Adepts: adaptive intrusion response using attack graphs in an ecommerce environment. Proc. of Dependable Systems and Networks, pages 508–17, 2005.
[11] S. Hsu and A. Arapostathis. Competitive Markov decision processes with partial observation. Proc. of IEEE lnt. Conf. on Systems, Man and Cybemetics, 1:236–41, 2004.
[12] L. Kaelbling, M. Littman, and A. Cassandra. Partially observable Markov decision processes for artificial intelligence. Proc. of the German Conference on Artificial Intelligence: Advances in Artificial Intelligence, 981:1–17, 1995.
[13] O. P. Kreidl and T. M. Frazier. Feedback control applied to survivability: A host-based autonomic defense system. IEEE Trans. on Reliability, 53:148–66, 2004.
[14] C. Kruegel, W. Robertson, and G. Vigna. Using alert verification to identify successful intrusion attempts. Info. Processing and Communication, 27:220–8, 2004.
[15] S. Lewandowski, D. Hook, G. O'Leary, J. Haines, and M. Rossey. SARA: Survivable autonomic response architecture. Proc. of the DARPA Info. Survivability Conf. and Exposition II, 1:77–88, 2001.
[16] M. Locasto, K. Wang, A. Keromytis, and S. Stolfo. FLIPS: Hybrid adaptive intrusion prevention. Proc. of the Symp. On Recent Advances in Intrusion Detection, pages 82–101, 2005.
[17] K. Lye and J. Wing. Game strategies in network security. Int'l Journal of Info. Security, 4:71–86, 2005.
[18] S. Musman and P. Flesher. System or security managers adaptive response tool. Proc. of the DARPA Info. Survivability Conf. and Exposition, 2:56–68, 2000.
[19] G. Owen. Game Theory. Academic Press, 1995.
[20] P. Porras and P. Neumann. EMERALD: Event monitoring enabling responses to anomalous live disturbances. Proc. Of the Info. Systems Security Conf., pages 353–65, 1997.
[21] D. Ragsdale, C. Carver, J. Humphries, and U. Pooch. Adaptation techniques for intrusion detection and intrusion response system. Proc. of the IEEE Int'l Conf. on Systems, Man, and Cybernetics, pages 2344–9, 2000.
[22] R. Rehman. Intrusion Detection Systems with Snort. Prentice-Hall, 2003.
[23] B. Schneier. Secrets & Lies: Digital Security in a Networked World. John Wiley & Sons, 2000.
[24] A. Somayaji and S. Forrest. Automated response using system call delay. Proc. of the USENIX Security Symp., pages 185–97, 2000.
[25] E. Sondik. The Optimal Control of Partially Observable Markov Processes. PhD Thesis: Standford University, 1971.
[26] N. Stakhanova, S. Basu, and J. Wong. Taxonomy of intrusion response systems. Int'l Journal on Info. And Computer Security, pages 169–84, 2007.
[27] A. Valdes and K. Skinner. Adaptive, model-based monitoring for cyber attack detection. Proc. of the Recent Advances in Intrusion Detection, pages 80–92, 2000.
[28] G. White, E. Fisch, and U. Pooch. Cooperating security managers: A peer-based intrusion detection system. IEEE Network, pages 20–3, 1996.