Providing Response to Security Incidents in the Cloud Computing with Autonomic Systems and Big Data

Kleber M. M. Vieira, Daniel S. M. Pascal Filho, Carlos B. Westphall, João Bosco M. Sobral, Jorge Werner
{kleber,westphal,bosco}@inf.ufsc.br, {jorge.werner,daniel.smpf}@gmail.com
LRG - INE - UFSC - Florianopolis - SC - Brazil

***Abstract*—This article provides a real-time intrusion response system in order to reduce the consequences of the attacks in the Cloud Computing. Our work proposes an autonomic intrusion response technique that uses a utility function to determine the best response to the attack providing self-healing properties to the environment. To achieve this goal, we propose the Intrusion Response Autonomic System (IRAS), which is an autonomic intrusion response system, using Big Data techniques for data analysis.**

Keywords–Cloud; Security; Intrusion Detection System; Big Data.

I. INTRODUCTION

As a complement to the work presented in [1], the object of this article is to present the results and details of its implementation. Because of their distributed nature, cloud computing environments are a great target for intruders interested in exploring possible vulnerabilities in their services and consequently using the abundant resources maliciously.

The growing number of attacks and vulnerability exploitation techniques requires preventative measures by system administrators. In this context, the need for a highly effective and rapid reactive security system gains importance. These measures are getting more complex with the growth of data heterogeneity and the increasing complexity of the attacks. In addition, slow reaction time from human agents and the huge amount of data and information generated, makes the decision making process an arduous task. In response to this, there is an increase in the usage of Intrusion Detection Systems (IDS) [2], as a way to identify attack patterns, malicious actions and unauthorized access to an environment [3].

The need for IDS is growing due to limitations in Intrusion Preventing Systems (IPS) - which focus on alerting administrators when a vulnerability is detected, connectivity and threat evolution, as well as the financial appeal of cybercrime [4].

Despite their growing importance, currently available IDS solutions have limited response mechanisms. While the research focus is on better intrusion detection techniques, response and effective threat reaction are still mostly manual and rely on human agents to take effect [5].

Recently, some intrusion detection tools have begun providing limited sets of automated responses, but with the growing complexity of intrusions, the need for more effective response system strategies has increased. Due to implementation limitations, research on intrusion detection techniques advance faster than intrusion response systems [3].

The development of reliable and rapid responsive systems is even more important for cloud computing, in which elasticity increases the risk and costs of an attack [6].

A. Motivation

The number of computer attacks has grown in quantity and complexity in the recent years, making defense an increasingly arduous task. Every computer that suffers an attack has very limited information on who initiated the attack and its origin. Current intrusion detection and response systems do not keep up with the growing number of threats [5].

The focus on manual processes creates a delay between detection and response, leaving a window of opportunity for attackers [7]. Research findings by Lumpur [5] indicate that if a skilled attacker has a period of 10 hours between intrusion and response, the attack has an 80% chance of success. If the attacker has 20 hours, the attack has a 95% chance of success, and at 30 hours the attack becomes virtually foolproof. In this situation, the system administrator's skills become irrelevant. On the other hand, if the response to the intrusion is immediate, the chance of a successful attack is almost zero. Lumpur says that statistics have shown that the number of pro-rated intrusions is growing. The high cost of a contract indicates serious financial commitment made by the Pentagon to prevent and secure their infrastructure from another country.

An automated intrusion response system that incorporates the best intrusion detection techniques would offer the best possible defense in a short time frame, affording the system administrator more time to develop a permanent solution to prevent future attacks or to fix the exploited vulnerability [5] [7].

According to Buyya [8], the Cloud is complex, extensive, heterogeneous, and challenging to manage. This environment requires an automated and intelligent system to provide cost-efficient security services. Thus, cloud systems represent a distinct structure, with several layers of abstraction, that requires specific IDS and response techniques to address its complexity.

B. Goals

In this article, we propose a model for autonomic intrusion detection system based on the autonomic loop, commonly referenced as MAPE-K (Monitor, Analyze, Plan, Execute and Knowledge Base). To monitor and analyze, we use sensors to collect data from IDS logs, network traffic, system logs, and data communication. For storage and further analysis, distributed storage is used. For instance, we chose Apache

 ISBN: 978-1-61208-411-4

Hadoop as a storage engine because of its performance, scalability and further capabilities to be extended and perform MapReduce jobs.

This paper is organized as follows: Section 2 describes the proposal's underlying concepts and key technologies. Section 3 presents an overview of the related work. Section 4 details the proposal. Section 5 show the results of execution tests. Section 6 concludes the paper with future goals and open challenges.

II. Autonomic Computing

Autonomic computing can overcome the heterogeneity and complexity of computing systems and is being considered a new and effective approach to implement complex systems, by addressing several areas in which humans are losing control due to system complexity and slow reaction time, such as the security systems area [9].

The autonomic computing model is based on the so called self properties. The self is inspired by the autonomic nervous system of the human body, which can manage multiple key functions through involuntary control. The autonomic computing system is the adjustment of software and hardware resources to manage its operation, driven by changes in the internal and external demands. It has four key features, including self-configuration, self-healing, self-optimization and self-protection.

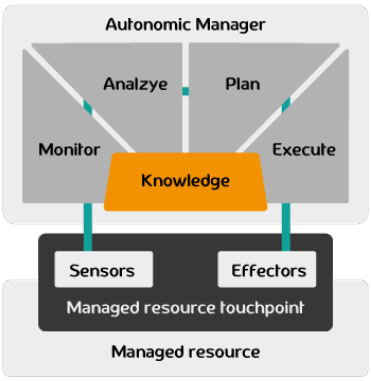

Figure 1. An autonomous system.

Figure 1 shows the structure of an autonomic system and its Monitor, Analysis, Planning, Executor and Knowledge (MAPE-K) cycle [10], composed by the Monitoring, Analysis, Planning and Executing modules. All the management of the autonomic component is performed by a meta-management element, which makes decisions based on the knowledge-base it built.

Sensors are responsible for collecting information from the managed element. Information collected by the sensors is sent to the monitors where they are interpreted, preprocessed, aggregated and presented in a higher level of abstraction. After this, the analysis phase is executed and planning takes place. At this stage, a work plan is created, which consists of a set of actions to be performed by the executor. Only the sensors and executors have direct access to the managed element. Through the autonomic management cycle, there may be a need for decision-making, and thus the presence of the knowledge base is also necessary [11].

A. Autonomic Systems Properties

The essence of autonomic computing is self-management. To implement it, the system must be self-aware as well as environment-aware. Thus, the system must precisely know its current situation and be aware of the operational environment in which it operates. From a practical standpoint, according to Hariri [11], the term autonomic computing has been used to denote systems that have the following properties:

- Self-awareness: the system knows itself, including its components, their state and behavior.
- Context-awareness: the system must be aware of the context of its execution environment and be able to react to changes in its environment.
- Self-configuring: the system must dynamically adjust its resources based on its status and the state of the execution environment.
- Self-optimizing: the system is able to detect performance degradations and functions to perform self-optimization.
- Self-protecting: the system is able to detect and protect its resources from external and internal attackers, maintaining its overall security and integrity.
- Self-healing: the system must have the ability to identify potential problems and to reconfigure itself in order to continue operating normally.

III. Related Work

In this section, five related papers that we considered important to our research were selected. To evaluate these, five topics were chosen to analyze them: focus on IDS, relation to the Cloud scenario, attack response, self-healing method, and algorithm used.

Chai [12] presents an in-flow event processing system for autonomic computing . This system is resistant to hardware failures and attacks. The mechanism votes on consuming events. It also introduces an evidence-based safe-guarding mechanism that prevents a faulty event.

Wu [13] proposes an autonomous manager which introduces a mechanism for multi-attribute auction. Its architecture has a layer of managed resources generically covering all physical devices such as routers, servers and software applications. These resources should be manageable, observable, and adjustable. The state of resources refers to all data (events) that reflect the state of existing resources, including logging and real-time events. This architecture also has an autonomous agent as a detection engine, optimization strategy, autonomic response, and a knowledge base module. Wu says that the autonomic response depends on a knowledge base of possible actions. It is necessary to create a knowledge base with attributes and valuations [13].

The Kholidy [14] approach describes how to extend the current technology and IDS systems. His proposal is based on a hierarchical IDS to experimentally detect DDoS, host-based, network based and masquerade attacks. It provides capabilities for self-resilience preventing illegal security event updates on data storage and avoiding single point of failure across multiple instances of intrusion detection components. Kholidy's proposal consists of a hierarchical structure, autonomic and Cloud based, extending his earlier work with features such as

 ISBN: 978-1-61208-411-4

autonomic response and prediction. In particular, it assesses vulnerabilities and risks in the system through a mechanism that builds a security model based on risk assessment and security event policies criticality. It also provides the possibility of an automatic response to actions based on a set of policies defined by the system administrator. However, a black box format does not clarify possible answers or make clear how to choose the best answer leaving that decision to a system administrator. Finally, the architecture offers some predictive capabilities based on Holt-Winters algorithm [15], which predicts and detects abnormal behavior in network traffic when the amount of collected network traffic is either too high or too low, compared to normal network traffic. Predictive capabilities improve detection accuracy of both decision making and automated response [16].

Vollmer [17] describes new architecture that uses concepts of autonomic computing, based on SOA and an external communication layer to create a network security sensor. This approach simplifies the integration of legacy applications and supports a safe, scalable, self-managed structure. The contribution of this piece is a flexible two level communication layer, based on autonomic computing and SOA. One module uses clustering and fuzzy logic to monitor traffic for abnormal behavior. Another module passively monitors network traffic and deploys deceptive hosts in the virtual network. This work also presents the possibility of an automatic response but it does not address the topic in detail, leaving it for future research.

Sperotto [18] presents an autonomic approach to adjust the parameters of intrusion detection systems based on SSH traffic anomalies. Sperotto proposes a procedure which automatically tunes system parameters, and in doing so, optimizes system performance. Their approach was validated by testing it on a probabilistic-based detection test environment for attack detection, on a system running SSH.

A. About the related works

Related papers representing the state of the art attempt to solve the problem of cyber-attacks by proposing intrusion detection mechanisms and increasing detection techniques. Although many of them show the need for automatic responses, none of them go further in this direction. The works of Wu [13] and Vollmer [17] mention the possibility of attack response. However, neither delves deeper into the issue.

Table I shows a brief comparison of the related works, based on the previously described topics.

IV. PROPOSAL

We propose an intrusion response autonomic system (IRAS) based on MAPE-K. Here we will explain each module of system.

A. Proposed system: IRAS Intrusion Responsive Autonomic System

The approach of IRAS follows the method of an autonomic system for intrusion response. The sensors collect log data from the network IDS and host systems. This information is compiled in a Big Data environment [19], preprocessed and placed in a higher level of abstraction, ready to be sent to the analysis and planning cycles of the autonomic loop.

Based on the MAPE-K autonomic loop, IRAS, as shown in Figure 2, their modules are:

- Monitor: data collection from sensors, and storage on Big Data infrastructure.
- Analysis: preprocessing (filtering, aggregation) and analysis.
- Planning: calculation of utility.
- Executor: based on results of the utility function, effective measures will be taken in the system.
- Knowledge: database, built from the monitored and analyzed data, is fed back into the utility based function, weighting the utilities.

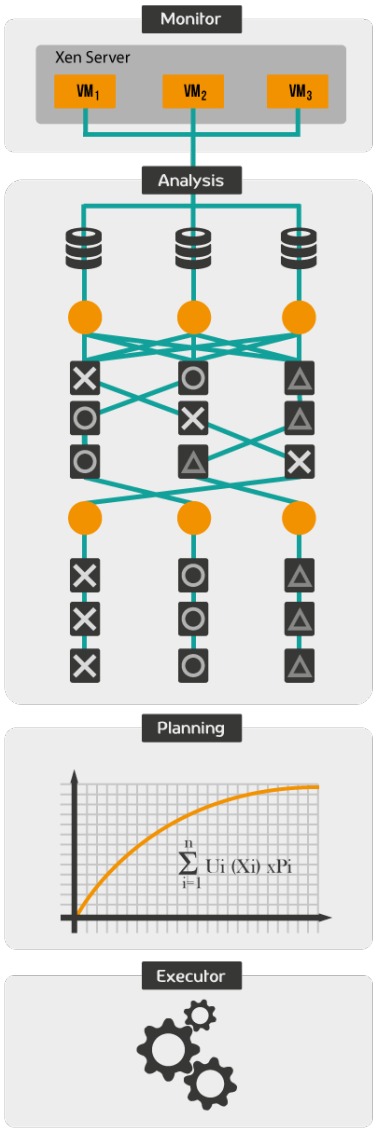

Figure 2. Intrusion Responsive Autonomic System IRAS

B. Monitoring

The first phase of the MAPE-K autonomic cycle corresponds to monitoring. In this step, sensors are used in order

TABLE I. RELATED WORKS

Author	IDS	Cloud	Response	Self-healing	Big Data	Algorithm
Wu	yes	no	yes	no	no	Auction
Kholidy	yes	yes	yes	no	no	Holt- Winters
Vollmer	yes	no	yes	no	no	Fuzzy
Sperotto	yes	no	no	no	no	Flor-based
Chai	yes	no	no	yes	no	Byzantine fault tolerance

to obtain data, reflecting changes in behavior of the managed element, or information from the execution environment that is relevant to the self-management process.

The concept of a sensor is a little generic, but it is possible to consider a sensor as a component of the system that makes the connection between the external world and the management system.

However, the important nuance to observe in data monitoring for security in Cloud Computing is that the data will be intrinsically temporal. This characteristic imposes some peculiarities in the data structure to store temporal information, as well as in the queries to be executed on the sensor database to retrieve useful information.

To monitor and analyze, we used sensors to collect data. It is important collect data from VMs and Hypervisor. Our monitor collects data from IDS logs in the Hypervisor and VMs, network traffic in the entire infrastructure, system logs, and data communication.

C. Analysis

The analysis phase queries the monitoring data looking for events that can characterize attacks.

As defined by Manyika [20], Big Data refers to datasets whose size is beyond the ability of typical database software tools to capture, store, manage, and analyze. Zikopoulos [21] defines the three data characteristics of Big Data sets: volume, variety and velocity. We have a large volume of data from various sources such as logs, IDS alerts, and network traffic scans, in which processing and analysis speed is necessary to extract meaningful information from these sources. Based on work by Suthaharan [19], we decided to use a structure with Big Data tools, which in this case was Hadoop, to organize the collected data in the Cloud and perform monitoring. However, Suthaharan uses Machine Learning (ML) to find attacks and in this paper we propose to use technical knowledge based on intrusion detection systems [2], making it possible to detect attacks like Stuxnet or Duqu. Thus we made a map reduced over the collected data to identify signatures of known attacks, by extracting significant data such as origin, destination of attack, type, signature and timestamp.

There is a resourceful set of analytics methods that correlates data in order to discover causality relationship, or events association. There are three types of analytical methods that are useful for Cloud Computing security:

- Diagnostic: this method means to synthesize a temporal flow of events arising from sensors in a *security state* of the Cloud - it is common to represent the state as a dashboard.
- Root-cause: the goal of this type of analysis is to determine what events are the main causes of the current Cloud state.
- Prediction: the prediction methods aims to suggest forecast projections to the Cloud state.

It is possible to consider that the analysis phase in Cloud Computing security management has the following characteristics:

- There must be evaluation methods able to supply a set of security metrics for parts of and for the entire Cloud.
- It must consider temporality – generally based on time series.
- It must be multi-criteria – there may be multiple, seemingly uncorrelated, events that, perceived together, constitute an attack.
- It must learn – the measures in a real world Cloud changes their statistical distribution, variance and behavior – in this context, an analytical method to security in Cloud Computing must be adaptive to follow these changes.

The root-cause analysis will not be addressed in this paper. However, it may be important to correlate and determine how some configuration states (e.g. a blocked ip address in the firewall) influence the occurrence of security incidents. In this way, a sensor component reads the data from logs, IDS agents, VM and Hypervisor [22] data collectors, network traffic sniffers, SNMP agents and alarms. This analysis will be important to determine and discover possible security actions.

The prediction will be important to establish the consequences of an action $a \in A$ execution, where A is the set of all possible actions, over a state $s \in S$. So, the prediction of action consequences must provide a probability function $p(s^{t+1}|a, s^t)$, read as: the probability of action a, executed over a state s^t in time t conduce to a state s^{t+1} in time $t+1$.

D. Planning

The Planning Phase receives events from the analysis phase and must choose one action to offer the autonomic system properties self-configuration, self-healing, self-optimization, and self-protection.

To carry out the planning, the Expected Utility technique was chosen [23].

E. Utility

Here we consider the use of utility to find the best response to the attacks. The utility function comes from economy studies (REF) and is expressed by the equation Ui (S). The states that have the best utility should be chosen.

The higher the U, the better. The utility function is expressed as follows:

$$U[x_1, x_2, x_3...x_n] = u_1(x_1)+u_2(x_2)+...u_n(x_n) = \sum_{i=1}^{n} u_i(x_i)$$

$$\max_{x \in D} u[x_1, x_2, x_3...x_n]$$

An example of the application of utility: Let us say that in a meal the utility of coffee is 1, orange juice, 2, bread, 3 and a cookie, 4. Thus, we can express the utility of breakfast by: U (drink, solid) = u. The option with the highest utility should be chosen, which in this case would be U (orange, cookie) = 6.

F. Expected Utility

Incrementing our utility function with the uncertainty that the response may block an attack and bring self-healing to the environment, we use the probability of the event [23].

$$UE[x_1...x_n] = u_1(x_1) \times p_1 + ...u_n(x_n) \times p_n = \sum_{i=1}^{n} u_i(x_i) \times p_i$$

$$\max_{x \in D} u[x_1, x_2, x_3...x_n]$$

For example, given a scan attack, one possible response is to block the source IP.

The probability of this event succeeding is 50%.

$$P(blockIP) = \frac{2}{1}.$$

If the value of the block IP action has a utility value of 5, we can express this as follows:

$$UE(blockIP) = 5 \times 0,5 = 2,5.$$

With the history of this response it is possible to over time optimize the environment, granting the self-healing autonomic property to the environment.

G. Executor

After calculating the response with the highest expected utility, it is possible to forward the response to an executing agent in the Cloud. The hypervisor is responsible for executing the response, providing transparence for each virtual machine.

As shown in Table 1, our work presents an increase in the state of the art when you use Big Data to locate attack occurrences and to be able to provide a response that takes into consideration the impacts of the attack across the Cloud environment. Regarding the authors Wu [13] and Vollmer [17], the contribution of our research was to consider the Cloud environment and the peculiarities of its hypervisor, and the complexity of providing a response without being invasive to customers. Our work also considers self-healing and uses a statistical function in expected utility to achieve the most efficient response and thereby, block the attacks.

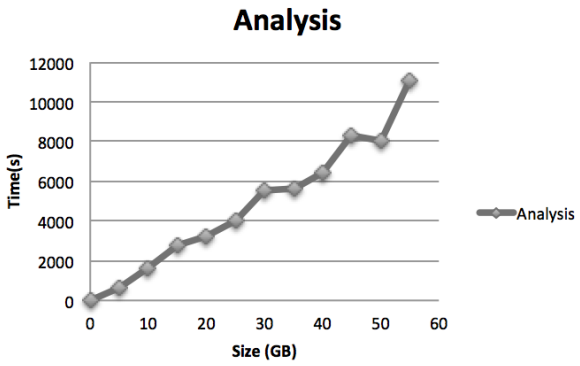

Figure 3. Analysis module execution

V. Execution

We developed a prototype to evaluate the architecture described in this work. It uses Cloudera, Xen Cloud and Cloud Stack. We use JnetPCap to capture network traffic and the parse data. Afterwards we used MapReduce to organize the data by source IP, transport layer and application layer. With organized data the search can be completed more efficiently. After this process is done, a search for known attacks is performed. Data tables are created to perform the experiments with audit elements coming from both the system log and data captured from network. We prepared two types of simulation data to perform the tests.

- Data representing legitimate actions: A set of known services simulating normal behavior was executed to prepare this type of data.
- Data representing knowledge attacks.

The Analysis and Planning module presented in this paper was implemented in Java. For the Analysis module, we used Hadoop. This module was the critical processing point. To perform the MapReduce, 1841 seconds were needed to process 10 GB. The results are shown in Figure 3. After the MapReduce, the result was a small table with data for the Planning module. In this test we used only one instance of the Cloud. To achieve improved performance we could create a larger number of instances.

VI. Conclusion

This paper proposed an autonomic computation system to respond to attacks. The current state of the art was researched and compared with the proposal described here. The solution was distributed into four main modules: Monitoring, Analysis, Planning and Execution. A prototype was presented, which, for the Monitoring module, captured all data transferred in the network. For analysis, we used the Big Data Hadoop tool. For the Planning module, in order to make the best attack response decisions the expected utility function was used, a technique inspired by economics. This solution makes it possible for the Cloud environment to have a self-healing capability against attacks. Tests were performed in the Cloud environment with a large volume of data. The results made it possible to detect attacks and provide a response to them. In this way a we created a self-healing property for the cloud environment. For future research, we suggest focusing on the need to improve the

 ISBN: 978-1-61208-411-4

performance of the Analysis module in order to have a greater efficiency of resource use, in relation to the large amount of data. It is also possible to use a resource limit criterion for the utility function, to get the best response, which uses fewer cloud computing resources.

REFERENCES

[1] K. M. Vieira, F. Schubert, G. A. Geronimo, R. de Souza Mendes, and C. B. Westphall, "Autonomic intrusion detection system in cloud computing with big data," in The 2014 International Conference on Security and Management (SAM 2014)., 2014.

[2] K. Vieira, A. Schulter, C. Westphall, and C. M. Westphall, "Intrusion detection for grid and cloud computing," It Professional, vol. 12, no. 4, 2010, pp. 38–43.

[3] N. Stakhanova, S. Basu, and J. Wong, "A taxonomy of intrusion response systems," International Journal of Information and Computer Security, vol. 1, no. 1, 2007, pp. 169–184.

[4] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in Cloud," Journal of Network and Computer Applications, vol. 36, no. 1, Jan. 2013, pp. 42–57. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S1084804512001178

[5] K. Lumpur, "An investigation and survey of response options for Intrusion Response Systems (IRSs)," 2010.

[6] P. Mell and T. Grance, "The nist definition of cloud computing (draft)," NIST special publication, vol. 800, no. 145, 2011, p. 7.

[7] C. A. Carver, "Intrusion response systems: A survey," Department of Computer Science, Texas A&M University, College Station, TX, 2000, pp. 77 843–3112.

[8] R. Buyya, R. Calheiros, and X. Li, "Autonomic Cloud computing: Open challenges and architectural elements," Emerging Applications of ..., Nov. 2012, pp. 3–10. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6407847 http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=6407847

[9] J. Kephart and D. Chess, "The vision of autonomic computing," Computer, vol. 36, no. 1, Jan. 2003, pp. 41–50. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1160055

[10] M. C. Huebscher and J. A. McCann, "A survey of autonomic computing—degrees, models, and applications," ACM Computing Surveys (CSUR), vol. 40, no. 3, 2008, p. 7.

[11] S. Hariri, B. Khargharia, H. Chen, J. Yang, Y. Zhang, M. Parashar, and H. Liu, "The autonomic computing paradigm," Cluster Computing, vol. 9, no. 1, 2006, pp. 5–17.

[12] H. Chai and W. Zhao, "Byzantine fault tolerant event stream processing for autonomic computing," in Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on. IEEE, 2014, pp. 109–114.

[13] Q. Wu, X. Zhang, R. Zheng, and M. Zhang, "An Autonomic Intrusion Detection Model with Multi-Attribute Auction Mechanism," vol. 10, no. 1, 2013, pp. 56–61.

[14] H. A. Kholidy, A. Erradi, S. Abdelwahed, and F. Baiardi, "Ha-cids: A hierarchical and autonomous ids for cloud systems," in Computational Intelligence, Communication Systems and Networks (CICSyN), 2013 Fifth International Conference on. IEEE, 2013, pp. 179–184.

[15] C. Chatfield, "The holt-winters forecasting procedure," Applied Statistics, 1978, pp. 264–279.

[16] H. Kholidy, A. Erradi, S. Abdelwahed, and F. Baiardi, "A hierarchical, autonomous, and forecasting cloud IDS," 2013, pp. 213–220.

[17] D. Vollmer, M. Manic, and O. Linda, "Autonomic Intelligent Cyber Sensor to Support Industrial Control Network Awareness," IEEE Transactions on Industrial Informatics, vol. PP, no. 99, 2013, pp. 1–1. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6547755

[18] A.-b. Idss, S. S. H. Case, A. Sperotto, M. Mandjes, R. Sadre, P.-t. D. Boer, A. Pras, and P.-T. de Boer, "Autonomic Parameter Tuning of Anomaly-Based IDSs: an SSH Case Study," IEEE Transactions on Network and Service Management, vol. 9, no. 2, Jun. 2012, pp. 128–141. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6172597

[19] S. Suthaharan, "Big data classification: Problems and challenges in network intrusion prediction with machine learning," in Big Data Analytics workshop, in conjunction with ACM Sigmetrics, 2013.

[20] J. Manyika, M. Chui, B. Brown, J. Bughin, R. Dobbs, C. Roxburgh, and A. H. Byers, "Big data: The next frontier for innovation, competition, and productivity," McKinsey Global Institute, May 2011.

[21] P. Zikopoulos, C. Eaton et al., Understanding big data: Analytics for enterprise class hadoop and streaming data. McGraw-Hill Osborne Media, 2011.

[22] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, and M. Rajarajan, "A survey of intrusion detection techniques in cloud," Journal of Network and Computer Applications, vol. 36, no. 1, 2013, pp. 42–57.

[23] R. F. Bordley and S. M. Pollock, "A decision-analytic approach to reliability-based design optimization," Operations research, vol. 57, no. 5, 2009, pp. 1262–1270.

 ISBN: 978-1-61208-411-4