Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 127 (2018) 35–41

Procedia Computer Science

www.elsevier.com/locate/procedia

www.elsevier.com/locate/procedia

The First International Conference On Intelligent Computing in Data Sciences

Distributed Intrusion Detection System for Cloud Environments based on Data Mining techniques

Mohamed Idhammad[a,*], Karim Afdel[a], Mustapha Belouch[b]

[a]*LabSIV, Department of Computer Science, Faculty of Science, Ibn Zohr University, BP 8106, Morocco*
[b]*LAMAI, Department of Computer Science, FSTG, Cadi Ayyad University, Morocco*

Abstract

Nearly two decades after its emergence, the Cloud Computing remains gaining traction among organizations and individual users. Many security issues arise with the transition to this computing paradigm including intrusions detection. Intrusion and attack tools have become more sophisticated defeating traditional Intrusion Detection Systems (IDS) by large amount of network traffic data and dynamic behaviors. The existing Cloud IDSs suffer form low detection accuracy, high false positive rate and high running time.

In this paper we present a distributed Machine Learning based intrusion detection system for Cloud environments. The proposed system is designed to be inserted in the Cloud side by side with the edge network components of the Cloud provider. This allows to intercept incoming network traffic to the edge network routers of the physical layer. A time-based sliding window algorithm is used to preprocess the captured network traffic on each Cloud router and pass it to an anomaly detection module using Naive Bayes classifier. A set of commodity server nodes based on Hadoop and MapReduce are available for each anomaly detection module to use when the network congestion increases. For each time window, the anomaly network traffic data on each router side are synchronized to a central storage server. Next, an ensemble learning classifiers based on the Random Forest is used to perform a final multi-class classification step in order to detect the type of each attack.

Various experiment are performed in the Google Cloud Platform in order to assess the proposed system using the CIDDS-001 public dataset. The obtained results are satisfactory when compared to a standard Random Forest classifier. The system achieved an average accuracy of 97%, an average false positive rate of 0.21% and an average running time of 6.23s.

Keywords: Intrusion Detection Systems, Cloud Computing, Machine Leaning, Hadoop, MapReduce

1. Introduction

The appealing features of Cloud computing continue to fuel its integration in many sectors including industry, governments, education, entertainment, to name few [1].

* Corresponding author. Tel.: +212653097813
E-mail addresses: idhammad.mohamed@gmail.com (Mohamed Idhammad)., k.afdel@uiz.ac.ma (Karim Afdel)., mbelouch@gmail.com (Mustapha Belouch).

10.1016/j.procs.2018.01.095

Cloud computing aims to provide convenient, on-demand, network access to a shared pool of configurable computing resources, which can be rapidly provisioned and released with minimal management effort or service provider interactions [2].

The pay-as-you-go and the on-demand elastic operation Cloud characteristics are changing the enterprise computing model, shifting on-premises infrastructures to off premises data centers, accessed over the Internet and managed by cloud hosting providers.

However, many security issues arise with the transition to this computing paradigm including intrusions detection. Regardless the important evolution of the information security technologies in recent years, intrusions and attacks continue to defeat existing intrusion detection systems in Cloud environments [3]. Attackers developed new sophisticated techniques able to brought down an entire Cloud platform or even many within minutes. New records are breached each year by attacker. Recently a destructive DDoS attack have brought down more than 70 vital services of Internet including Github, Twitter, Amazon, Paypal, etc. Attackers have taken advantages of Cloud Computing and Internet of Things technologies to generate a huge amount of attack traffic; more than 665 Gb/s [4, 5].

Intrusion and attack tools have become more sophisticated challenging existing Cloud IDSs by large volumes of network traffic data, dynamic and complex behaviors and new types of attacks. It is clear that an IDS for Cloud should analyze large volumes of network traffic data, detect efficiently the new attack behaviors and reach high accuracy with low false. However preprocessing, analyzing and detecting intrusions in Cloud environments using traditional techniques have become very costly in terms of computation, time and budget.

Therefore, efficient intrusions detection in Cloud environments requires adoption of new distributed and intelligent techniques such as Machine Learning techniques.

In this paper we present a distributed Machine Learning based intrusion detection system for Cloud environments. The proposed system is designed to be inserted in the edge network components of the Cloud provider. This allows to intercept incoming network traffic to the edge network routers of the Cloud. A time-based sliding window algorithm is used to preprocess the captured network traffic on each Cloud router and pass it to an anomaly detection module using Naive Bayes classifier. A set of commodity server nodes based on Hadoop and MapReduce are available for each anomaly detection module to use when the network congestion increases. For each time window, the anomaly network traffic data on each router are synchronized to a central storage server. Next, an ensemble learning classifiers based on the Random Forest is used to perform a final multi-class classification step in order to detect the type of each attack.

The proposed system is implemented in the Google Cloud Platform which use Hadoop and MapReduce to distribute and to scale computations over available clusters.

MapReduce is a processing technique and a programing model for distributed computing based on the programming language Java. MapReduce algorithms consist of two principal parts or classes Map and Reduce. The Map class takes a set of data and converts it into another set of data, where individual elements are broken down into tuples (key/value pairs). The Reduce class takes the output from a map as an input and combines those data tuples into a smaller set of tuples. As the sequence of the name MapReduce implies, the reduce task is always performed after the map task [6, 7]. Hadoop is a Cloud Computing platform capable of providing real-time data analysis with ease. The framework is widely used for exploration, processing and analysis of structured and non-structured data [8, 9, 10].

The remainder of this paper is organized as follows. Section 2 gives an overview on the related works. Section 3 is devoted to the dataset used in this paper. A thorough explanation of the proposed IDS is presented in section 4. Section 5 gives the details of the conducted experiments and the performance metrics used to evaluate the proposed IDS. The results and discussion are given in Section 6. Finally, this paper ends with the main conclusion.

2. Related works

This section introduces some previous works that are devoted to improve intrusion detection performances in Cloud environments.

Roschke et al. [11] have proposed a traditional IDS systems based on Snort named as VM-Integrated IDS to detect anomalies. The proposed IDS is an extensible architecture that consists of several sensors and a central management unit. The IDS combines system level virtualization technology and VM Monitoring approaches to handle VM-based IDSs. This allows basic capabilities to easily integrate the system in Cloud architectures.

Similarly, Modi et al. [12] used Snort and machine learning classifiers to detect anomalies in the network traffic between Vms. The framework proposed by authors integrates a network intrusion detection system (NIDS) in the Cloud infrastructure. It is based on Snort and decision tree (DT) classifier. It aims to mainly detect network attacks in the Cloud by monitoring network traffic. The authors used the NSL-KDD and KDD datasets to validate their system.

Gupta et. al [13] proposed a new IDS for a Cloud environment based on immediate Syscall signature structure to determine malicious program executions in Cloud. The proposed system is efficient in terms of complexity and resources consumption. Authors have evaluated the system in a private Cloud environment based on open nebula and virtual box using several datasets available on UNM (University of New Mexico). High accuracy was achieved for several attacks detection.

In [14] authors have applied Artificial Neural Network (ANN) to detect attacks in the cloud. They proposed a distributed neural network based IDS with an adaptive architecture that allows it to avoid overloading Cloud VMs. ANN allows the IDS to detect new types of attacks with fairly accurate results. The system was evaluated using the KDD dataset on a physical cloud testbed and have achieved satisfactory attacks detection results.

Mohamed I. et al. [15] have proposed a supervised DoS detection method based on a feed-forward neural network. This method consists of three major steps: (1) Collection of the incoming network traffic, (2) selection of relevant features for DoS detection using an unsupervised Correlation-based Feature Selection (CFS) method, (3) classification of the incoming network traffic into DoS traffic or normal traffic. The approach achieves good performances on the UNSW-NB15 and NSL-KDD datasets.

Mustapha B. et al [16] have presented a two-stage classifier based on RepTree algorithm and protocols subset for network intrusion detection system. The first phase of their approach consists of dividing the incoming network traffic into three type of protocols TCP, UDP or Other. Then classifying it into normal or anomaly traffic. In the second stage a multi-class algorithm classify the anomaly detected in the first phase to identify the attacks class in order to choose the appropriate intervention. Two public datasets are used for experiments in this paper namely the UNSW-NB15 and the NSL-KDD.

Gul et. al [17] have proposed a Cloud NIDS to detect network intrusions in Cloud environments. The system detects network attacks targeting tenant VMs based on rule-matching techniques. Also, the proposed Cloud IDS can handle large flow of data packets and analyze them efficiently.

In [18] an Hypervisor based IDS for Cloud environments is proposed. In the system the traditional misuse and anomaly detection techniques are integrated with VM introspection to improve the performances of the IDS in the cloud.

Most of the proposed IDS in Cloud environments are based on traditional techniques and IDS such as Snort IDS. Despite that important performances are achieved for standard information system infrastructures or private Cloud, they are limited face to the new distributed and sophisticated attacks. Hence, further scalable and distributed techniques should be adapted for Cloud IDSs.

3. CIDDS-001 dataset

CIDDS-001 (Coburg Intrusion Detection Data Set) dataset is an up-to-date labeled flow-based dataset created by M. Ring et. al [19] in a Cloud environment based on OpenStack platform. This environment includes several clients, emulated using a set of Python scripts, and typical servers including E-Mail server, Web server, etc. The dataset contains realistic normal and attack traffic allowing important benchmarking of network intrusion detection systems in a Cloud environment. The dataset is divided into four parts each is created during a week. The CIDDS-001 is a flow-based format dataset containing unidirectional NetFlow [19] data. The dataset contains 14 attribute, the first 10 attribute are the default NetFlow attributes and the last four attributes are additional attributes, names and descriptions of features are tabulated in table 1. A total of 32 million of normal and attack flows are captured in the dataset within four weeks. Authors have exploited 92 types of attacks to create the dataset. This makes the dataset an important benchmark for intrusion detection systems. The reasons above motivated us to use this dataset in this paper. In our experiments we used only the first week, CIDDS-001-week1. This part contains 8,451,520 instances of normal and attack traffic data.

Table 1: Features of the CIDDS-001 dataset

Feature	Description
1. Src IP	Source IP Address
2. Src Port	Source Port
3. Dest IP	Destination IP Address
4. Dest Port	Destination Port
5. Proto	Transport Protocol (e.g. ICMP, TCP, or UDP)
6. Date first seen	Start time flow first seen
7. Duration	Duration of the flow
8. Bytes	Number of transmitted bytes
9. Packets	Number of transmitted packets
10. Flags	OR concatenation of all TCP Flags
11. Class	Class label (normal, attacker, victim, suspicious or unknown)
12. AttackType	Type of Attack (portScan, dos, bruteForce, —)
13. AttackID	Unique attack id. All flows which belong to the same attack carry the same attack id.
14. Attack Description	Provides additional information about the set attack parameters (e.g. the number of attempted password guesses for SSH-Brute-Force attacks)

4. The proposed intrusion detection system

In this section the methodology followed to detect intrusions in the Cloud as well as the components of the proposed intrusion detection system are given.

4.1. Components of the proposed IDS

The proposed intrusion detection system is composed of the following components:

Cloud network traffic data collector module: implemented in the Cloud network controller close to each edge network router of the Cloud provider. This allows to capture incoming network traffic data to the Cloud services hosted on the same infrastructure. The module performs the capture in a 5 minutes time window basis. For each time window the captured network traffic data is stored in a storage server.

Network traffic data preprocessing module: preprocesses the captured network traffic data during each time window. Several techniques are used to format, clean and normalize the network traffic data. This module is designed to be inserted near to each Cloud network traffic data collector module. A cluster of commodity servers based on Hadoop and MapReduce is available to use by the preprocessing models when the network traffic congestion increase. The preprocessing tasks are implemented in MapReduce and deployed on the Hadoop cluster. To normalize the network traffic data features the *MinMax* method is used. In *MinMax* the values of features are scaled to the range [0, 1] as follows:

$$x_i^{new} = \frac{x_i - \min(X)}{\max(X) - \min(X)} \tag{1}$$

Where X is a feature of the network traffic data, x_i is the current value of X to normalize and x_i^{new} is the normalized value.

Anomaly detection module: classifies the network traffic data of each time window into normal or abnormal traffic. A Naive Bayes model built from the CIDDS-001 dataset is used to detect malicious traffic. This module is deployed side by side to each Cloud network traffic data preprocessing module which allow to speed up the anomaly detection with minimum resources.

Network traffic synchronization module: synchronizes the malicious network traffic of each Cloud router side to a centralized server. The synchronization is performed for each time window separately.

Attacks traffic classification module: classifies the malicious network traffic data synchronized to the central storage server. An ensemble learning classifiers based on Random Forest is used to performs a multi-class classification on the malicious network traffic data. This allows to detect the type of each attack and early perform convenient mitigation actions for it. The module classifies the available network traffic on the central storage server based on a FIFO time queue.

4.2. Entire architecture of the proposed IDS

This section introduces the process followed by the proposed IDS to detect attacks. For each time window the incoming network traffic data to the Cloud is captured at each one of the edge routers. The captured data are then preprocessed and passed to a first anomaly detection step using a Naive Bayes model. This allows to detect and block suspected traffic while allowing access to the normal traffic. The suspected traffic at each network router side are synchronized to a central server. An ensemble learning classifier based on Random Forest is used to classify the network traffic data available on the central storage server and detect the types of each attack. The intrusions detection process and the flowchart of the entire proposed IDS are illustrated in figure 1.

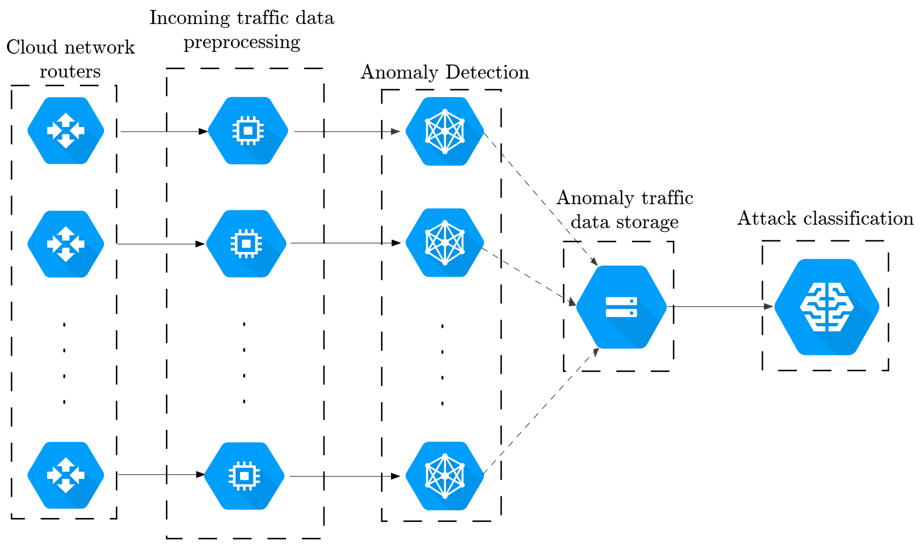
Incoming traffic data preprocessing
Cloud network routers
Anomaly Detection
Anomaly traffic data storage
Attack classification

Fig. 1: Flowchart of the proposed IDS.

5. Experiments

To assess the proposed intrusion detection system several experiments were performed on the CIDDS-001 dataset. First the dataset is split into train and test sets using a configuration of 60% for training and 40% for testing. In order to simulate the intrusions in the Cloud platform a 5 minutes time-based window sampling method is applied to the test set. The traffic records of each time window are divided subsequently to 4 parts, where each records go to a single router. At each one of the routers sides the preprocessing tasks and the anomaly detection are performed. Then the Random Forest ensemble learning is used to detect types of each intrusion. The obtained results of the entire proposed IDS are compared with a standard Random Forest ensemble classifiers tested directly on the CIDDS-001 dataset. The experiments testbed used in this paper constitutes of 4 instances created in the Google Cloud Platform with a total of 8 cores and 32 Go of memory.

6. Results and Discussion

6.1. Evaluation metrics

The proposed IDS classifies the network traffic data as either positive or negative which correspond respectively to an attack type or normal traffic. The obtained results are evaluated using the following performance metrics:

Accuracy: Percentage of the traffic records that are correctly classified.

$$Accuracy = 100 \times \frac{correctly\ \ classified\ \ records}{total\ \ records} \tag{2}$$

False Positive Rate (FPR): Percentage of the normal records which are classified as attack records.

$$FPR = 100 \times \frac{FP}{FP + TN} \tag{3}$$

Where FP is the number of normal records incorrectly classified as attack records. TN is the number of correctly classified normal records.

Running time: total time needed to detect intrusions including preprocessing time, anomaly detection time and Random Forest classification time.

ROC and AUC curves: Receiver Operator Characteristic (ROC) and Area Under ROC (AUC) curves are commonly used to present results for binary decision problems in machine learning. The ROC curve shows how the number of correctly classified positive examples varies with the number of incorrectly classified negative examples. The AUC value characterizes the accuracy of the model.

6.2. Performance of the proposed IDS

In this section we give the obtained results of the proposed IDS evaluated using the CIDDS-001 dataset. The obtained results of the anomaly detection module illustrated in figure 2. The results represent the ROC curves and the AUC scores of the anomaly detection module at each router side of the Cloud platform. The anomaly detection module achieves its highest accuracy of 94.3% at the third router. Also, the module achieves an accuracy of 89%, 92.7% and 91.4% for respectively routers 1, 2 and 4. Low running time and high false positive rates are also observed at each router.

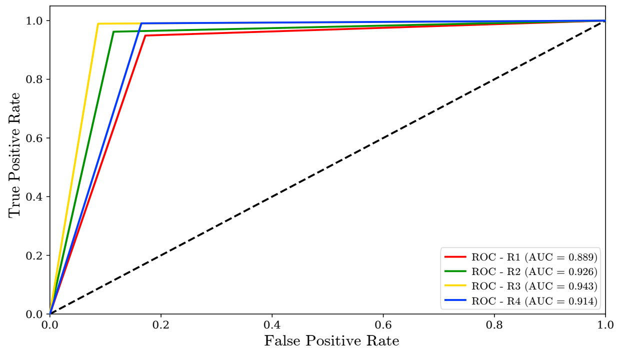

Fig. 2: ROC curves and AUC scores of the anomaly detection module of the proposed IDS at each router side.

The remaining obtained overall results of the proposed IDS are tabulated in table 2. The results in table 2 represents the average accuracy and the average false positive rates of the proposed IDS to detect the four types of attack classes contained in the dataset used in this paper which are DoS, PortScan, PingScan and BruteForce. The overall accuracy of the proposed IDS reached 97% for all the attack classes, with an average FPR of 0.21%. It worth nothing that the proposed IDS outperform the standard Random Forest tested directly on the CIDDS-001 dataset which achieved and accuracy of 85% with a high FPR of 0.43%. Also the proposed IDS achieved good running time of 6.23s for all the dataset compared to the standard Random Forest that achieved 24.87s.

Table 2: Overall performances of the proposed distributed IDS

Attack	avg Accuracy (%)	avg FPR (%)
DoS	99.2	0.12
PortScan	96.7	0.24
PingScan	97.8	0.22
BruteForce	94.5	0.26
Overall	97.05	0.21

7. Conclusion

In this paper a distributed intrusion detection system for Cloud environments is proposed. The proposed IDS constitutes of 5 principal modules. The network traffic module capture the incoming network traffic to the Cloud on each one of the edge network routers in a 5 minutes time window basis. The captured data are then preprocessed and passed to a first anomaly detection step using a Naive Bayes model. Next, the suspected traffic at each network router side are synchronized to central server. Then, an ensemble learning classifier based on Random Forest is used to classify the network traffic data available on the central storage server and detect the types of each attack.

The proposed IDS is implemented on the Google Cloud Platform and tested using the CIDDS-001 public dataset. The experimental results are satisfactory when compared to a standard Random Forest tested directly on the dataset.

Despite, that the proposed IDS depicts high detection performances for several attack types included in the CIDDS-001 public dataset, it is important to evaluate its performances in real world scenarios. For future work, we are planning to perform real world deployment of the IDS and evaluate it against several attack types.

References

[1] D. A. Fernandes, L. F. Soares, J. V. Gomes, M. M. Freire, P. R. Inácio, Security issues in cloud environments: a survey, International Journal of Information Security 13 (2) (2014) 113–170.

[2] P. Mell, T. Grance, The nist definition of cloud computing.

[3] S. Iqbal, M. L. M. Kiah, B. Dhaghighi, M. Hussain, S. Khan, M. K. Khan, K.-K. R. Choo, On cloud security attacks: A taxonomy and intrusion detection and prevention as a service, Journal of Network and Computer Applications 74 (2016) 98–120.

[4] Wikipedia, 2016 dyn cyberattack[Online; accessed 10-November-2017)].

[5] theguardian, Ddos attack that disrupted internet was largest of its kind in history, experts say[Online; accessed 10-April-2017)].

[6] D. Miner, A. Shook, MapReduce Design Patterns: Building Effective Algorithms and Analytics for Hadoop and Other Systems, " O'Reilly Media, Inc.", USA, 2012.

[7] J. Dean, S. Ghemawat, Mapreduce: simplified data processing on large clusters, Communications of the ACM 51 (1) (2008) 107–113.

[8] A. Holmes, Hadoop in practice, Manning Publications Co., Greenwich, CT, USA, 2012.

[9] J. Venner, S. Wadkar, M. Siddalingaiah, Pro Apache Hadoop, Apress, USA, 2014.

[10] Apache. The apache software foundation, apache hadoop 2.5.1, [online] (2008).

[11] S. Roschke, F. Cheng, C. Meinel, Intrusion detection in the cloud, in: Dependable, Autonomic and Secure Computing, 2009. DASC'09. Eighth IEEE International Conference on, IEEE, 2009, pp. 729–734.

[12] C. Modi, D. Patel, B. Borisanya, A. Patel, M. Rajarajan, A novel framework for intrusion detection in cloud, in: Proceedings of the Fifth International Conference on Security of Information and Networks, ACM, 2012, pp. 67–74.

[13] S. Gupta, P. Kumar, An immediate system call sequence based approach for detecting malicious program executions in cloud environment, Wireless Personal Communications 81 (1) (2015) 405–425.

[14] Z. Li, W. Sun, L. Wang, A neural network based distributed intrusion detection system on cloud platform, in: Cloud Computing and Intelligent Systems (CCIS), 2012 IEEE 2nd International Conference on, Vol. 1, IEEE, 2012, pp. 75–79.

[15] M. Idhammad, K. Afdel, M. Belouch, Dos detection method based on artificial neural networks, International Journal of Advanced Computer Science and Applications(ijacsa) 8 (4). `doi:http://dx.doi.org/10.14569/IJACSA.2017.080461`.

[16] B. Mustapha, E. H. Salah, I. Mohamed, A two-stage classifier approach using reptree algorithm for network intrusion detection, International Journal of Advanced Computer Science and Applications(ijacsa) 8 (6). `doi:http://dx.doi.org/10.14569/IJACSA.2017.080651`.

[17] I. Gul, M. Hussain, Distributed cloud intrusion detection model, International Journal of Advanced Science and Technology 34 (38) (2011) 135.

[18] V. Varadharajan, U. Tupakula, Security as a service model for cloud environment, IEEE Transactions on network and Service management 11 (1) (2014) 60–75.

[19] M. Ring, S. Wunderlich, D. Grdl, D. Landes, A. Hotho, Flow-based benchmark data sets for intrusion detection, in: Proceedings of the 16th European Conference on Cyber Warfare and Security (ECCWS), ACPI, 2017, pp. 361–369.