International Conference on Business Research
University of Moratuwa, Moratuwa, Sri Lanka
June 26, 2019.

BUSINESS RESEARCH UNIT
FACULTY OF BUSINESS
UNIVERSITY OF MORATUWA

NIDS BASED RANDOM MODEL TO PROTECTED BIG DATA ENVIRONMENT USING SPARK

S. Prema[1] and S. Asokkumar[2]

[1]Associate Professor, Information Technology

Mahendra Engineering College

Email: sprema074@gmail.com

[2]Professor and Head, Management Studies

Mahendra Engineering College

Email: asokkumar777@gmail.com

ABSTRACT

Big Data is an active business across the world. With the growing size of data comes many challenges connected with handing out and ensuring the security of huge data. In this paper, we propose a Network Intrusion Detection System (NIDS) model based Random Forests (RF) classifier for anomaly detection of the collected network traffic. In order to decrease the computational time connected with the bulk of captured data, we utilize the system of Hadoop, MapReduce and Spark that have proven to be among the most efficient and fault-tolerant systems. We use the NSL KDD cup 99 dataset to perform experimental analysis and Non-dominated Sorting Genetic Algorithm-II (NSGA-II) for feature selection over this dataset.

Keywords—Big data, NIDS, NSGA-II, Random Forests, Spark, Hadoop, MapReduce

1. Introduction

Big Data is massive heterogeneous data stored over commodity hardware, data centers, Cloud and many other storage devices. Heterogeneous sources and multiple technologies contribute to the production of data for a Big Data environment, including Wireless Sensor Networks (WSN), Internet of Things (IoT), Near-Field Communications (NFC), Cloud, social networking websites, black-box data, and so on (Dietrich, Heller and Yang, 2015). Real-time applications that are being developed using Big Data analysis to serve the live-ware also require management, security and governance of the analyzed data. Factors affecting the management of Big Data are built up on 5 Vs that are – Volume, Variety, Value, Velocity and Veracity (Dietrich, Heller and Yang, 2015; Moreno, Serrano and Fernandez-Medina, 2016).

However, Big Data is more to storage, as multifarious companies, medical industries, government, sports, military, space stations and many other such exciting fields are using Big Data analytics to identify certain patterns, heuristic studies, and relationships in various entities. Big Data analysis has helped many industries to save billions of dollars. With the advent of Big Data analysis comes several challenges regarding the safety and security of data, as well as computational complexities. From the aspects of futuristic applications, this

© 2019 BRU, University of Moratuwa

data will grow and traditional techniques would not be able to handle and secure such a huge amount of vital data.

Various security solutions such as firewalls, Intrusion Detection Systems (IDS), tokenization of data, data de-duplication are available in the market that provide infrastructure security, network security, data integrity, confidentiality of information and reactive security (Wang and Jones 2017). However, rogue nodes can introduce malicious data into the network that can lead to the exploitation of resources. Intrusion Detection Systems (IDSs) facilitate security for the entire infrastructure, data and network. IDS has two types; Anomaly-based IDS and Signature-based IDS. Since Signature-based IDS are not capable of detecting novel attacks, we choose Network Intrusion Detection Systems (NIDSs) that are capable of securing the entire network infrastructure along with novel attack detection (Zuech, Khoshgoftaar and Wald, 2015; Hasan, Nasser, Pal and Ahmad, 2014). Most of the NIDSs follow rule-based systems, for which their entire performance depends mainly on the rule-sets (Zhang, Zulkernine and Haque, 2008). While dealing with Big Data, the huge volume of network traffic generated from heterogeneous sources becomes difficult to manage using a rule-based system and it is time-consuming as well. Therefore, research has come up with data mining techniques which are in use over the past recent years and are proven to have significant advantages in securing the network infrastructure in a Big Data environment (Chen et al, 2016)). Random Forests (RF) were introduced by Breiman in 2001, which considers regression and classification problems for two-class and multi-class (Breiman, 2001).

Tackling gazillions of data entries and then further analyzing them to retrieve crucial information is not an easy task. Nowadays, Main Frames and supercomputers which are known for high performance computing are being used by some renowned and technical organizations. There are diverse solutions to deal with the challenges associated with Big Data and such systems are popularly known as Big Data Systems (BDSs). Hadoop and MapReduce (Patel, Birla and Nair, 2012) are such BDSs that resolve computational issues of a Big Data ecosystem. Apache Spark (Zaharia et al, 2016) is an advancement over Hadoop and MapReduce which performs computations 100 times faster in RAM and up to 10 times faster processing of data in storage. In addition, real-time data analytics can be done using Spark streaming. Components of a Spark ecosystem are shown in Figure 1 (Zaharia et al, 2016; Apache Spark 2017).

Keeping in mind the end goal to enhance performance in terms of accuracy, detection rate and computational speed, we propose a NIDS model using RF classifier to distinguish between incoming legitimate and malicious network traffic in a Big Data environment. Our proposed scheme utilizes Non-dominated Sorting Algorithm-II (NSGA-II) (Deb, Pratap, Agarwal and Meyarivan, 2002; Tamimi, Naidu and Kavianpour, 2015) for identifying and selecting the most promising features amongst the entire feature set for the detection of attacks. Besides, we utilize Hadoop, MapReduce and Spark collaboratively for reducing the computational complexity while dealing with bulk instances of incoming network traffic. We use NSL KDD cup 99 dataset (Dhanabal and Shantharajah, 2015) to execute and prove the correctness of our model.

The rest of the paper is organized as follows: Section 2 highlights the related work. Section 3 discusses the proposed model in detail including the rudimentary details of the concepts involved. Section 4 presents the security and performance analysis of the proposed model, along with a brief comparison with existing schemes. Finally, Section 5 concludes the paper with future work.

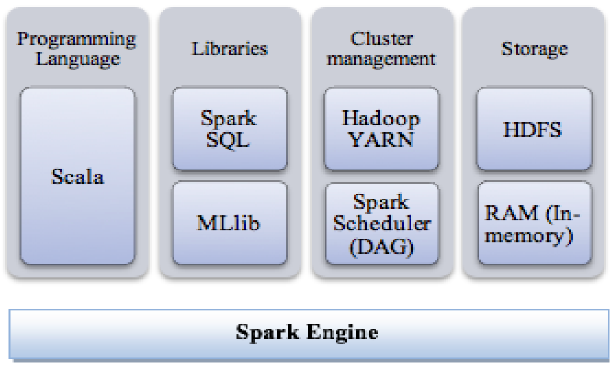


Figure 1: Apache Spark Components

2. Related Work

Genuer et al. (2017) analyzed the performance of RF with respect to the Big Data scenario by implementing its idea with five variants on two massive datasets with 15 and 120 million observations, respectively. The authors considered only sequential inputs. They also highlighted some of the limitations of this approach along with some suggestions to improve the overall efficiency of an RF based system. Chen et al. (2016) proposed a Parallel Random Forests (PRF) algorithm for Big Data environment which is developed using a combination of data parallel and task parallel optimization approaches in order to optimize the PRF. They implemented the algorithm on the Apache Spark platform. A dimension-reduction approach in the training process and a weighted voting approach in the prediction process prior to parallelization is performed as well to improve PRF's accuracy for large, high-dimensional and noisy data.

Sun et al. (2014) implemented their improved approach of feature selection for Random Forests on Spark system utilizing UCI dataset. They studied two important issues of feature selection that include – elimination of noisy features which have no relevance with classification, and elimination of redundant features. Zaharia et. al (2016) highlighted the importance of Apache Spark as one of the most powerful BDS which is used for scalable data processing along with the parallel computation. Spark facilitates the faster processing of Big Data according to the experimentation results of the authors. The concept of composability is utilized by Apache Spark which also inspired the development of conveniently interoperable libraries.

Deb et al. (2002) propounded the advantages of using NSGA-II for selecting the most prominent features from the dataset in order to evaluate the experimental results. Three vital issues that include high computational complexity of non-dominated sorting, need for specifying the sharing parameter and lack of elitism are taken into consideration while implementing NSGA-II. Tamimi et al. (2015) also proposed a model using NSGA-II which utilizes multi-objective functionality and generates rules for IDS. They used DARPA dataset to execute the experiments of the proposed model and to prove its correctness.

Zhang et al. (2008) used Random Forests which is a data mining algorithm for anomaly detection and also assists in detecting novel attacks. They evaluated their approaches using Knowledge Discovery and Data Mining 1999 (KDD'99) dataset which achieves higher detection and lower false alarm rates.

3. Proposed Work

a. Preliminaries

The most significant components of our proposed model are Hadoop, MapReduce, Spark, RF classifier, NSGA-II, and NSL KDD cup 99 dataset. Figure 2 is the illustration of our proposed model depicting the use of Spark's machine learning libraries, Hadoop and MapReduce in a Big Data environment and incorporation of NIDS using RF classifier. A brief description of these components is given as follows:

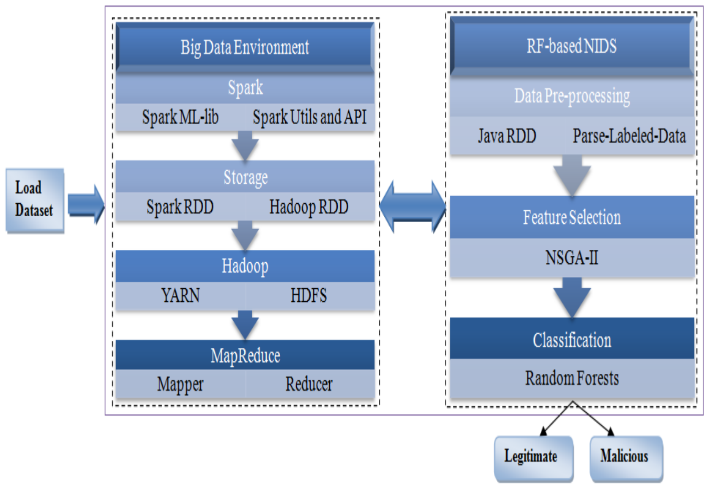


Figure 2: The Proposed Model

1. **Hadoop MapReduce (Patel et al, 2012; Del Rio et al, 2014) –** Hadoop is commonly known as "Big Data Handler". We use Hadoop 2.6.0 for the execution of our model. Hadoop has incorporated several distinct strategies, for example, Hadoop Distributed File System (HDFS), MapReduce, Hive, Pig, Zookeper, and so on. In specific circumstances, the data is required instantly. For this reason, data is kept in HDFS. Hadoop is capable of storing data in peta-bytes and zeta-bytes with no constraints on capacity, and performs faster calculation and computation over data when utilized as a part of joint effort with MapReduce. MapReduce is a programming model in Hadoop, which first separates data into autonomous pieces that are processed altogether by map task in parallel and later shuffled by the system. At that point, the output is given as contribution to the reduced task, which further reduces it to generate a final output.

2. **Apache Spark (Zaharia et al, 2016, Gupta and Kulariya, 2016] –** Spark is indeed an advancement over Hadoop as it is capable of processing data 100 times faster than Hadoop. It is a BDS built on Hadoop and runs using Spark Engine. We use Apache Spark version 2.1.0. to execute our experiment. Spark can be standalone or can work

in collaboration with the framework of MapReduce and HDFS, out of which we use the latter one. Spark's components include SQL, Streaming, GraphX, Machine Learning library (MLlib), and so on. There are only a few classifiers supported by Spark that we use in our proposed model as classifiers and regression models in Apache Spark are still evolving.

3. **Spark RDD (Chen et al, 2016) –** Apache Spark supports a programming model which is very similar to MapReduce, however extends it with "Resilient Distributed Datasets" (RDD) which facilitates fault-tolerance environment. Using this extension, Spark is able to capture a broader range of processing workloads that previously needed separate engines, including streaming, machine learning, SQL, and graph processing.
4. **Data Preprocessing (Hamed, Ernst and Kremer, 2017) –** We use NSL KDD cup 99 dataset which is already pre-processed to some extent. We further pre-process it using Parse-labeled-point. In MLlib, labeled points are used in supervised learning algorithms. We use a double to store a label, along with the conversion of string values to integers in order to utilize it in regression and classification. Spark.ml package provides machine learning APIs built on the data frames that are also becoming the core part of Spark SQL library. Spark.ml package is used for developing and managing the machine learning pipelines, feature extraction, as transformers and selectors.
5. **NSGA-II [Deb et al, 2002; Tamimi et al, 2015; Kaliappan, Thiagarajan and Sundararajan, 2015] –** We use Information Gain (IG) [20] and NSGA-II for feature selection phase. We apply the methodology likewise in the proposed framework of Kaliappan et al. (2015) to calculate IG values. Feature selection is done to filter out the less promising and irrelevant features for NIDS in our NSL KDD cup 99 dataset. We incorporate the upgraded version of NSGA for feature selection i.e. NSGA-II which uses fast elitist NSGA and is best suited for the Big Data environment. Table 1 represents the NSGA-II parameters adjusted according to the requirement of our proposed model.

Table 1: NSGA-II Parameters

Modeling Description	Setting
Population Size	40
Crossover Rate	0.5
Mutation Rate	0.1
Binary Chromosome Length	41
Crowding Distance Sorting	Dynamic
Pareto Front	Dynamic
Non-Domination Rank	Dynamic

6. **NIDS (Hasan et al, 2014; Gupta and Kulariya, 2016) –** Examining the incoming/outgoing traffic, one can determine whether it is DoS attack. Probe, R2L (Remote-to-Local) and U2R (User-to- Root) attacks can also be identified using this technique. Anomaly-based IDS is used for the detection of malicious packets which are apparently the content of a network and need to be filtered out from legitimate packets or data. We train our classifiers using supervised learning in a manner so that we can incorporate them for a real time Big Data environment as well. Information regarding classification of different attack patterns for different attack types in the dataset used is provided in Table 2 (Hasan et al, 2014).

Table 2: Attack pattern available in NSL KDD 99 dataset for different attack types

Attack Type	Attack Pattern
Probe	ipsweep, nmap, portsweep, satan, mscan, saint
DoS	back, land, neptune, pod, smurf, teardrop, apache2, worm, udpstorm, mailbomb, processtable
U2R	buff_overflow, loadmodule, rootkit, perl, sqlattack, xterms, ps
R2L	guess_password imap, multihop, phf, spy, warezclient, warezmaster, ftp_write, xlock, xsnoop, snmpgue, snmpgetattack, httptunnel, sendmail, named
Normal	Normal

7. **Random Forest (RF) (Zhang et al, 2008; Breiman, 2001; Tamimi et al, 2015) –** RF is a supervised classification algorithm which is similar to bootstrapping algorithms that grow bootstrap samples from the original data and an unpruned classification tree to predict the new data by aggregating the predictions of the new tree. Using RF in collaboration with MapReduce for a Big Data scenario produces better results in terms of accuracy, Detection Rate (DR) and False Alarm Rate (FAR) (Kaliappan et al, 2015). In RF algorithm, maximum trees must be selected to have a dense forest, which leads to better classification results. We use RF classifier available in Apache Spark with all the essential libraries and adjusted parameters keeping in mind the end goal which is depicted in Table 3.

Table 3: Parameter setting for RF

Parameters of RF	Values
Number of Tress	15
Maximum Depth	8
String impurity	gini
Maximum Bins	32
Seed (bootstrapping)	20000

8. **NSL KDD 99 Dataset (Dhanabal and Shantharajah, 2015) –** This dataset is a newly proposed dataset which overcomes the issues of existing KDD Cup 99 dataset, such as, it does not include redundant records in the training set and there is no duplicity of records in the test sets. Apparently, we can say that it is a reduced version of the KDD cup 99 dataset. There are 125,973 records in the training set and 22,544 records in the test set, with 41 attributes ranging from 0 to 40. Table 4 represents the types of attacks available in NSL KDD 99 dataset along with their number of instances in Train and Test file.

Table 4: Types of attacks in NSL KDD 99 dataset

Attack Type/Class	Train File	Test File
DoS	45927	7456
Probe	11656	2421
R2L	995	2756
U2R	52	200
Normal	9711	9711
Total Instances	125973	22544

b. System Working

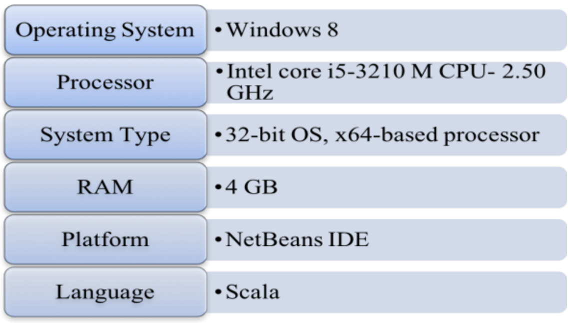
Once, we have acclimated our environment to execute our experiment with the configured system, we are prepared to consolidate all the system entities alluded to in the preliminaries section. Our first step is to run components of Apache Spark, which initiates Hadoop YARN and MapReduce on a NetBeans IDE platform. Once we have imported all the required libraries of Spark including HDFS lest we desire to execute our experiment on a local host, we prepare our platform to execute the next three important steps from providing input to obtaining output.

Firstly, we perform data pre-processing using Java RDD and parse-labeled-data that are provided by Java and Scala on the NSL KDD cup 99 dataset. This step lets us work with our dataset with no setbacks of redundancy, string conversion issues, and chaotic adjustments of instances. We utilize all the attacks in combination with Normal (Legitimate packets), for example; Normal-DoS which consists of all the Normal packets and DoS consists of all the packets that fall in the category of DoS based on their attributes. Secondly, we employ NSGA-II to perform feature selection on our dataset utilizing the parameters as mentioned in Table 1. Table 5 represents the most promising features selected by NSGA-II which assisted us in procuring the enhanced performance in respect with accuracy, detection rate and processing speed.

Eventually, the experiment executes with the incorporation of the classification procedure that embodies RF classifier for the same and provides output in two categories – Attack and Not-Attack. The RF classifier utilizes parameters which are mentioned in Table 3.

4. Performance Evaluation and Results

Prior to conducting experiment, we set up the environment with the system's configuration as shown in Figure 3. Table 5 outlines the features that are selected by NSGA-II in a dynamic fashion to meet the specific end goal in terms of higher accuracy and DR with respect to combination of each attack type with Normal or legitimate instances.



Operating System	• Windows 8
Processor	• Intel core i5-3210 M CPU- 2.50 GHz
System Type	• 32-bit OS, x64-based processor
RAM	• 4 GB
Platform	• NetBeans IDE
Language	• Scala

Figure 3: System Configuration

Table 5: Total promising features selected by NSGA-II dynamically using which maximum accuracy is achieved

Normal-Attack	**Selected Features**
Normal-DoS	0, 1, 3, 4, 5, 6, 7, 8, 9, 10, 14, 15, 16, 17, 19, 20, 22, 23, 24, 26, 27, 28, 30, 31, 32, 36, 37, 39, 40
Normal-Probe	0, 2, 3, 4, 6, 8, 10, 11, 12, 14, 15, 16, 19, 20, 21, 24, 26, 27, 28, 29, 30, 32, 35, 37, 38
Normal-R2L	0, 1, 2, 3, 6, 7, 8, 9, 10, 11, 13, 14, 15, 18, 20, 21, 22, 23, 24, 25, 26, 28, 32, 33, 34, 35, 37, 38, 40
Normal-U2R	0, 1, 2, 3, 4, 6, 8, 9, 12, 13, 15, 16, 17, 18, 19, 21, 23, 24, 25, 27, 28, 30, 32, 33, 34, 35, 36, 37

Figure 4 shows the graph depicting the results associated with accuracy, precision and the F-measure for each Normal-Attack combination which is considered with and without feature selection. For DoS, Probe and R2L attack categories, values are higher while considering feature selection using NSGA-II. On contrary, there is a slight variation in case of U2R, where higher values are achieved without considering feature selection, because of less instances. Table 6 shows the values for FAR and DR for each combination of Normal-Attack type with and without feature selection. We achieved high performance in terms of accuracy, DR and processing speed (testing time) by incorporating our proposed model for securing Big Data environment. Table 7 shows the comparison of the results of the proposed model with other such similar techniques.

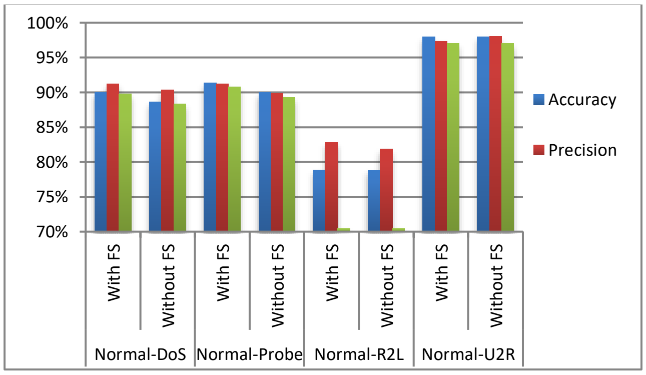


Figure 4: Measurement of Accuracy, Precision and F-Measure

Table 6: Results for false alarm rate (FAR) and detection rate (DR)

Normal-Attack	**Feature Selection (FS)**	**FAR**	**DR**
Normal-DoS	With FS	0.21	0.99
	Without FS	0.25	0.99
Normal-Probe	With FS	0.35	0.98
	Without FS	0.40	0.97
Normal-R2L	With FS	0.99	0.99
	Without FS	0.99	1.00
Normal-U2R	With FS	0.95	0.99
	Without FS	0.95	0.99

Table 7: Comparative analysis of different approaches that used RF classifier for NIDS

Approaches	**Model Description**	**DR**	**FAR**	**Testing Time (sec)**
Kaliappan et. al. (2015)	Multiple IDS fusion-based model	99	1-1.38	NA
Zhang et. al. (2008)	RF-based NIDS	94.7	2 %	NA
Hasan et. al. (2014)	RF modeling for IDS	NA	FNR 49.45	637.2
Proposed Model	RF and NSGA-II based model	99	0.2-0.9	21

5. Conclusion and Future Work

The growing pace with which data is being generated and transferred across the digital networks is building up a more cognitive computing environment. However, the issues related to security and processing of this bulk data, known as Big Data, are also proliferating. In this paper, we proposed an NIDS model based on RF classifier to detect the attack instances in network traffic. We used the NSL KDD cup 99 dataset to perform experimental analysis and NSGA-II algorithm for feature selection on Hadoop and Spark platform.

For future work, this execution can be carried out with massive data. Moreover, this entire execution can be done on Flink which is faster than Apache Spark. Additionally, Big Data analytics can be brought into consideration for analyzing the malicious traffic, user behaviors, virus signatures, timing of incident events and website profiles.

References

Apache Spark (2017). https://spark.apache.org/. Accessed on December 2017.

Breiman, L. (2001). "Random forests," *Machine learning*, vol. 45, pp. 5-32.

Chen, J., Li, K., Tang, Z., Bilal, K., Yu, S., Weng, C. and K. Li, (2016). "A parallel random forest algorithm for big data in a Spark cloud computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, pp. 919-933.

Deb, K., Pratap, A., Agarwal, S. and T. A. M. T. Meyarivan, (2002). "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE transactions on evolutionary computation*, vol. 6, pp. 182-197.

Del Río, S., López, V., Benítez, J. M. and F. Herrera, (2014). "On the use of MapReduce for imbalanced big data using Random Forest," *Information Sciences*, vol. 285, pp. 112-137.

Dietrich, D., Heller, B. and B. Yang, (2015). "Data Science and Big Data Analytics: Discovering," Analyzing, Visualizing and Presenting Data.

Dhanabal, L. and S. P. Shantharajah, (2015). "A study on NSL-KDD dataset for intrusion detection system based on classification algorithms," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 4, pp. 446-452.

Genuer, R., Poggi, J.M., Tuleau-Malot, C. and N. Villa-Vialaneix, (2017). "Random forests for big data," *Big Data Research*, vol. 9, pp. 28-46.

Gupta, G. P. and M. Kulariya, (2016). "A framework for fast and efficient cyber security network intrusion detection using apache spark," *Procedia Computer Science*, vol. 93, pp. 824-831.

Hamed, T., Ernst, J. B. and S. C. Kremer, (2017). "A Survey and Taxonomy on Data and Pre-processing Techniques of Intrusion Detection Systems," *Computer and Network Security Essentials*, pp. 113-134.

Hasan, M. A., Nasser, B. Pal, and S. Ahmad, (2014). "Support vector machine and random forest modeling for intrusion detection system (IDS)," *Journal of Intelligent Learning Systems and Applications*, vol. 6, p. 45.

Kaliappan, J., Thiagarajan, R. and K. Sundararajan, (2015). "Fusion of heterogeneous intrusion detection systems for network attack detection," *The Scientific World Journal*, July 2015.

Moreno, J., Serrano, M. A. and E. Fernández-Medina, (2016). "Main issues in big data security," *Future Internet*, vol. 8, p. 44.

Patel, A. B., Birla, M. and U. Nair, (2012). "Addressing big data problem using Hadoop and Map Reduce," *2012 Nirma University International Conference on Engineering (NUiCONE)*, Ahmedabad, India, pp. 1-5.

Sun, K., Miao, W., Zhang, X. and R. Rao, (2014). "An improvement to feature selection of random forests on spark," *2014 IEEE 17th International Conference on Computational Science and Engineering (CSE)*, Chengdu, China, pp. 774-779.

Tamimi, A., Naidu, D.S., and S. Kavianpour, (2015). "An Intrusion Detection System Based on NSGA-II Algorithm," *2015 Fourth International Conference on Cyber Security, Cyber Warfare, and Digital Forensic* (CyberSec), Jakarta, Indonesia, pp. 58-61.

Wang, L. and R. Jones, (2017). "Big data analytics for network intrusion detection: A survey," *International Journal of Networks and Communications*, vol. 7, pp. 24-31.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., M. Armbrust et al., (2016). "Apache spark: a unified engine for big data processing," *Communications of the ACM*, vol. 59, pp. 56-65.

Zhang, J., Zulkernine, M. and A. Haque, (2008). "Random-forests-based network intrusion detection systems," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, pp. 649-659.

Zuech, R. Khoshgoftaar, T. M. and R. Wald, (2015). "Intrusion detection and big heterogeneous data: a survey," *Journal of Big Data*, vol. 2, p. 3.