Performance Comparison For Multi Class Classification Intrusion Detection In SCADA Systems Using Apache Spark

Raogo Kabore[1], Yvon Kermarrec[2] and Philippe Lenca[3]

***Abstract*— As more and more data are generated by industrial control systems, the use of big data analysis approaches is of high interest today. In this paper we compare the performance of Apache Sparks classification algorithms i.e. Naive Bayes, Random Forest, Decision Tree, Multilayer Perceptron, for a multi-class intrusion detection using a SCADA dataset. The performance measure criteria used are the recall, precision, specificity, f-score, training time and prediction time.**

I. INTRODUCTION

Supervisory Control and Data Acquisition (SCADA) systems enable control and monitoring of systems such as water distribution systems, wastewater treatment systems, power transmission and distribution systems, oil and gas pipelines, public transport systems,etc. Those systems collect information from remote sites, transfer them to a central computer, and a Human Machine Interface (HMI) allows operators to monitor and control the entire system from a central facility location [1]. This integration of SCADA systems into the management of industrial systems can improve performance and reduce operating costs[3], but at the same time exposes those systems to the same attacks faced by traditional computing. Historically, SCADA systems used proprietary hardware and protocols and relied on two forms of protection i.e Air gap and security through obscurity. The Air Gap is based on the fact that SCADA networks are physically isolated from other networks making any attack difficult. The second form of protection is security through obscurity which relies on the presumption that information about SCADA systems were not available to public, thereby making them secured [4]. No consideration was given to information security in the design of SCADA systems. But nowadays, modern SCADA systems use Commercial-Off-The-Shelf (COTS) hardware and software, and standard communication technologies such as TCP/IP or Wireless protocols. Moreover, todays SCADA systems are interconnected to corporate networks and to the Internet for diverse reasons such as management, system administration etc. All the above shift in SCADA systems though allowing easy management and reduction of costs exposes them to attacks, mainly through cyber means [5]. The consequences of an attack on SCADA systems can be loss of production, financial losses, environmental disasters and even loss of life [6],[7] . SCADA networks have already been the targets of attacks such as the Maroochy Water System in Australia which spills wastewater in nearby rivers causing pollution[8] , the Davis-Besse nuclear power plant in Ohio (USA) attack shuts down the plant for five hours. Duqu and Flame are highly specialized malware used for espionage[9]. Stuxnet was the first ever discovered military-grade malware used to sabotage the iranian uranium enrichment centrifuges [10]. Attacks targeting power grids caused severe power outage in Ukraine [11] and Vermont in the USA [12].

SCADA systems are different from traditional computing because of the specificity of their protocols, the deterministic nature of the communications and the ill-adapted use of software patches. Those systems are real-time, require high availability, have a high number of legacy hardware and components with limited processing capacity and memory. They also have a static topology [13], [14]. Many approaches have been proposed to secure the SCADA networks such as cryptography, anti-viruses, software patches, firewalls, etc., but the specificity of the SCADA systems exposed above make the traditional IT solutions not always effective. Intrusion Detection systems (IDS) are therefore complementary solutions to secure SCADA systems.

The particular nature of SCADA systems require specific approaches for SCADA intrusion detection systems. In the present work, we analyze the use of big data approaches to make an effective multi-class anomaly detection in SCADA systems. Apache Spark which is a distributed, fast, in-memory, fault-tolerant is used in our project as the processing engine. The ML Machine Learning library of Spark contains a large set of algorithms. The Random Forest, Naive Bayes, Decision Trees and Multilayer Perceptron approaches are used for the multi-class intrusion detection in the water tank storage SCADA network dataset. Our objective is to make a comparison of the previous approaches in detecting different classes of attacks in a SCADA dataset using recall, specificity, precision, training time and detection time criteria. In section II of the paper, we present the related works, followed by section III in which we give details on the proposed approach. In Section IV, we present the experimental results obtained before concluding and giving future research directions in section V.

II. RELATED WORKS

Kulariya et al. [15] propose a performance analysis of Apache Spark's Logistic Regression, Support Vector Ma-

[1]R. Kabore is with IMT Atlantique Lab-STICC UMR CNRS 6285 UBL, Technopole Brest Iroise, F-29238 Brest cedex r.kabore at imt-atlantique.fr

[2]Y. Kermarrec is with IMT Atlantique Lab-STICC UMR CNRS 6285 UBL, Technopole Brest Iroise, F-29238 Brest cedex y.kermarrec at imt-atlantique.fr

[3]P. Lenca is with IMT Atlantique Lab-STICC UMR CNRS 6285 UBL, Technopole Brest Iroise, F-29238 Brest cedex p.lenca at imt-atlantique.fr

chine, Naive Bayes, Random Forest and Gradient Boosted Decision Trees intrusion detection techniques. They used KDD'99 cup dataset to compare the Accuracy, Sensitivity, Specificity, Prediction time end Training time measures of those techniques. However, this study does not provide a performance of the approaches on different classes of attacks in the KDD'99 dataset. Ahmed et al. [16] analyze the performance of intrusion detection techniques using Nearest Neighbor, clustering and statistical approaches. Their work focuses on using various SCADA Big Data datasets in order to compare the detection algorithms using False Positive Rate, Recall, F-1 and MCC measures, however, they don't deal with multi-class classification issues. Beaver et al. [17] did a mix of binary and multi-class classification performance evaluation on Naive Bayes, Random Forest OneR, J48, NNge, SVM in the detection of attacks in critical infrastructures, but this work is not using big data approaches. Using the DARPA' 98 dataset, Lazarevic et al. [18] made a comparison between the Local Outlier Factor (LOF), unsupervised SVM, Nearest Neighbor and Mahalanobis approaches in the detection of certain types of networks attacks. They are using different measures from those used in our study as burst detection rate, surface area and response time. Belouch et al. [19] use accuracy, sensitivity and specificity to compare the efficiency of Naive Bayes, Decision Tree, SVM, and Random Forests approaches available in Apache Spark. They use for that purpose the newly available dataset UNSW-NB15 containing normal and synthesized attacks networks traffic tuples. However, they are not addressing the problem of multi-class classification in their analysis.

III. PROPOSED APPROACH

A. CLASSIFICATION ALGORITHMS

For this study, we have selected Apache Spark which is an open source big data processing platform. Spark is backed by a strong community has built-in machine learning libraries with various algorithms for Classification, Regression and Clustering. For our performance analysis, we use Decision Tree, Random Forests, Naive Bayes and Multilayer Perceptron classifiers .

1) Decision Trees: A Decision tree is composed of three elements [20]:
- a node specifying a test attribute.
- an edge or a branch corresponding to the test attribute outcome.
- a leaf which represents the object's class.

In the training phase, a training dataset is used to build the tree by selecting for each node the appropriate attribute and defining the class label for each leaf. To classify a new instance, we start by a test on the root node attribute. The result of the test is use to move down a branch satisfying the test, and the process is repeated until a leaf is encountered. The label of the leaf is therefore the class of the new instance. Classification And Regression Tree (CART), ID3 and C4.5 are among the most popular Decision Tree algorithms. The majority of those algorithms uses a descendant strategy which requires an attribute selection measure parameter. Taking into account the discriminative power of each attribute over classes in order to choose the best one as the root of the (sub) decision tree. In other words, this measure should consider the ability of each attribute A_k to determine training objects classes.

2) Random Forests: A Random Forest (RF) [21] [22] [23] is an ensemble of decision trees. Each tree in the forest is built and tested independently from other trees, hence the overall training and testing procedures can be performed in parallel. During the training, each tree receives a new bootstrapped training set generated from the original training set by subsampling with replacement. We refer to those samples which are not included during the training of a tree as the Out-Of-Bag (OOB) samples of that tree. These samples can be used to compute the Out-Of-Bag-Error (OOBE) of the tree as well as for the ensemble which is an unbiased estimate of the generalization error.
During training, each decision node of the tree creates a set of random tests and then selects the best among them according to some quality measurement (*e.g.*, information gain or *Gini index*). The trees are usually grown to their full size without pruning.

3) Naive Bayes: Naive Bayes [20] are very simple Bayes networks which are composed of directed acyclic graphs (DAGs) with only one root node (called parent), representing the unobserved node, and several children, corresponding to observed nodes, with the strong assumption of independence among child nodes in the context of their parent. The classification is ensured by considering the parent node to be a hidden variable stating to which class each object in the testing set should belong and child nodes represent different attributes specifying this object. Hence, in presence of a training set we should only compute the conditional probabilities since the structure is unique. Once the network is quantified, it is possible to classify any new object giving its attributes values using the Bayes rule. expressed by:

$$P(c_i|A) = \frac{P(A|c_i).P(c_i)}{P(A)}$$

where c i is a possible value in the session class and A is the total evidence on attributes nodes.

4) Multilayer Perceptron: Multilayer Perceptron [25] is an Artificial Neural Network algorithm. A weighted sum of signal arriving at the neuron which is the basic unit of the Multilayer Perceptron is subjected to a *transfer function*. Several different transfer functions like *Sigmoid, threshold, linear* or *piece-wise linear* can be used.

The neurons are arranged in output layer and hidden layers. There is no communication between neurons of the same layer and adjacent layers are fully interconnected.
Let's w_{ji} be the weight of the link from the j-th hidden

neuron to the i-th output neuron and w_{kj} the weight of the link from the k-th attribute to the j-th hidden neuron.
In *forward propagation mode*, if a tuple $X = (x_1, x_2, ..., x_2)$ is presented to the network (See Figure 1), its attributes are passed along the links to the neurons. The values x_k being multiplied by the weights associated with the links, the j-th hidden neuron receives as input the weighted sum, $\sum_k w_{kj}x_k$, and subjects this sum to the sigmoid, $f(\sum_k w_{kj}x_k)$. The i-th output neuron then receives the weighted sum of the values coming from the hidden neurons and, again, subjects it to the transfer function. This is how the i-th output is obtained. The process of propagating in this manner the attribute values from the networks input to its output is called forward propagation.

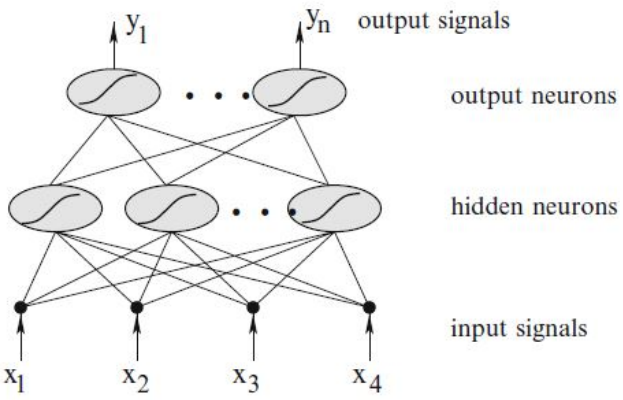

Fig. 1. Two interconnected layers neural network

B. PERFORMANCE MEASURES

Accuracy, recall (or sensitivity), precision, specificity, training time and prediction time [26], [27] are the measures we have selected to evaluate the performances of the Machine learning algorithms.

Positive tuples are the tuples of main interest in a dataset.

In an intrusion detection system where we are detecting anomalies, the positive tuples are the abnormal ones.

Negative tuples are all the other tuples.

Let P be the number of positive tuples and N the number of negative tuples.

True Positive (TP) represents positive tuples that were correctly labeled by the classifier.
True Negative(TN) represents negative tuples that were correctly labeled by the classifier.
False Positive (FP) represents negative tuples that were incorrectly labeled as positive.
False Negative (FN) represents positives tuples that were incorrectly labeled as negative.

Accuracy (or recognition rate)

$$Acc = \frac{TP + TN}{P + N}$$

Error rate (or misclassification rate)

$$Err = 1 - Acc = \frac{FP + FN}{P + N}$$

For imbalanced datasets (i.e main class of interest rare), the Accuracy measure is not acceptable. Other metrics such as sensitivity, specificity, recall are used instead.

Sensitivity or recall
The recall is the True Positive rate. It gives the proportion of positive tuples correctly identified.

$$Sensitivity = recall = \frac{TP}{P}$$

Specificity
Specificity is the True Negative rate i.e the proportion of negative tuples correctly identified.

$$Specificity = \frac{TN}{N}$$

Precision
Precision is a measure of exactness.It answers to the question *What percentage of tuples labeled as positive are actually such?*

$$Precision = \frac{TP}{TP + FP}$$

We are also using two speed measures that represent the computational cost involved in generating and using a given model :

Training duration : time took by a given algorithm to train the training dataset and generate a model.

Detection duration : time took by a model to predict the classes of testing dataset instances.

C. THE SCADA DATASET USED

The dataset used is obtained from a testbed of the Mississippi State University SCADA Security Laboratory and Power and Energy Research laboratory which is representing a water storage tank system [29]. The records of the dataset were captured from the control system of the water storage tank that models oil storage tanks found in industries like chemical or refineries [28]. This dataset contains 28 attacks against the Modbus Industrial Control System that monitors the water storage tank. Those attacks are grouped into four categories : reconnaissance attacks, command injection attacks, response injection attacks and denial-of-service (DoS) attacks. Moreover, some of those categories are split into subcategories, yielding to an overall eight categories for the dataset records as shown in Table I, including seven attack categories and one normal category.

TABLE I

DATASET RECORDS CATAGORIES

Label	Category	Description
0	Normal	Instance not part of an attack
1	NMRI	Naive Malicious Response Injection attack
2	CMRI	Complex Malicious Response Injection attack
3	MSCI	Malicious State Command Injection attack
4	MPCI	Malicious Parameter Command Injection attack
5	MFCI	Malicious Function code Command Injection attack
6	DoS	Denial-of-Service attack
7	Reconnaissance	Reconnaissance attack

D. INTRUSION DETECTION SYSTEM FRAMEWORK

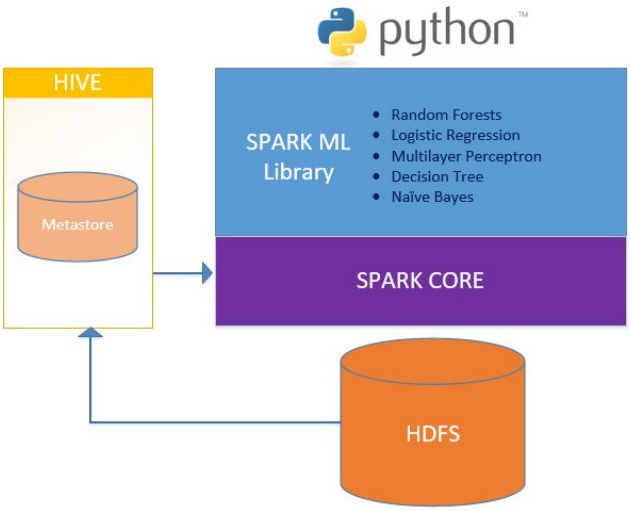

Fig. 2. SCADA IDS Solution design

The general framework of our intrusion detection system (Figure 2) is using Apache Spark ML API with the PySpark (Python for Spark) language. Hadoop HDFS is used to store the raw dataset and Apache Hive is used to enable the conversion of raw dataset into dataframe within Spark. The process of the intrusion detection and performance measure is as follow : The water tank storage raw dataset in flat file (.arff format) is first stored in Hadoop HDFS file system. Next, the data is ingested into Hive as a Hive table. Then with PySpark, the data is loaded on the Cluster as a Dataframe. At this step, the data should have been partitioned and sent to the clusters nodes (the workers) for a parallel processing, but in our setup, we are working locally. The dataset which has 236.179 records is split into training data and testing data (70 Naive Bayes) and a model or classifier is generated as the outcome of this process. The testing data is used to evaluate the performance of the generated models by checking their predictive power by the mean of different metrics (accuracy, recall, precision, specificity, f-score, training time and prediction time).

IV. EXPERIMENTAL RESULTS

A. Development environment

For the experimentations, we use a Virtual Private Server (VPS) with 3 Intel(R) Xeon(R) CPU E5-2680 0 @ 2.70GHz processors, 8 GB of memory and 40 GB of disk.
The VPS is running a Ubuntu 16.04.3 LTS OS. Hadoop 2.7.3, Apache Hive 2.3.0 and Apache Spark 2.0.0 were used for the development. PySpark (Python for Spark) is the programming language used.

B. Results

We are using the recall (or sensitivity), precision, Specificity, f-score, training time and prediction time metrics to compare Decision Tree, Random Forest, Logistic Regression, Multilayer Perceptron and Naive Bayes algorithms.
This is a multi-class classification (8 classes) as we have 7 categories of attacks (1 through 7) and category 0 for normal tuples.
The f-score or f1 is the harmonic mean of recall and precision. It can be used in some situations in lieu of recall and precision. But in this study, we focus our analysis on recall, precision and specificity.

TABLE II

DECISION TREE CLASSIFIER

Class	Count	Recall	Precision	Specificity
0	51365	1.000000	1.000000	1.000000
1	2804	1.000000	1.000000	1.000000
2	3730	1.000000	1.000000	1.000000
3	533	1.000000	1.000000	1.000000
4	1142	0.982487	1.000000	1.000000
5	351	1.000000	0.474966	0.994476
6	368	0.000000	0.000000	1.000000
7	10294	1.000000	1.000000	1.000000
Training time		7.841 s		
Prediction time		0.234 s		

1) Decision Tree: The Decision Tree classifier has a very good recall and precision for all classes of attacks except class 6 corresponding to the Denial-of-service (DoS) attacks. With a precision of 0, it's mean that no DoS attack is detected by the Decision Tree classifier.
We can also note that the classifier has 47 % of precision at detecting class 5 attacks which correspond to Malicious Function code Command Injection (MFCI). This situation means that there is a lot of False Positives in the detection of those attacks, the False Negatives being almost non-existent as show by the 99.45 % of specificity (True Negatives rate) . The classifier is fast to train (7.841 s) and also has a good prediction time (0.234 s).

2) Random Forest: The Random Forest algorithm does good prediction for all classes except class 6. But in this case, with a recall of 60 %, it means that only 60 % of the Denial-of-service attacks are detected leading to 40 % of

TABLE III
RANDOM FOREST CLASSIFIER

Class	Count	Recall	Precision	Specificity
0	51365	1.000000	1.000000	1.000000
1	2804	0.987874	0.999278	0.999970
2	3730	0.999464	0.949084	0.997009
3	533	0.956848	0.984556	0.999886
4	1142	0.987741	0.991213	0.999856
5	351	1.000000	1.000000	1.000000
6	368	0.600543	1.000000	1.000000
7	10294	1.000000	1.000000	1.000000
Training time		24.731 s		
Prediction time		0.344 s		

False Negatives. Compared to Decision Tree, the training time is slow (24.731 s) but the prediction time is rather good (0.344 s).

TABLE IV
MULTI LAYER PERCEPTRON CLASSIFIER

Class	Count	Recall	Precision	Specificity
0	51365	0.995892	0.920781	0.771044
1	2804	0.966120	0.917372	0.996400
2	3730	0.000000	0.000000	0.998340
3	533	0.000000	0.000000	0.999943
4	1142	0.990368	0.679688	0.992325
5	351	0.000000	0.000000	1.000000
6	368	0.000000	0.000000	1.000000
7	10294	1.000000	0.999417	0.999900
Training time		155.513 s		
Prediction time		0.157 s		

3) Multi-layer Perceptron: Our Multilayer Perceptron which is an Artificial Neural Network of two layers (1 output layer and 1 hidden layer) mis-classifies category 2 (Complex Malicious Response Injection), category 3 (Malicious State Command Injection), category 5 (Malicious Function Command Injection) and category 6 (Denial-of-Service) attacks. Out of eight classes, four have a recall of 0 (0 True Positive detected). We can conclude that Multilayer Perceptron performs poorly in this SCADA dataset multi-class classification. However, we should underline that this classifier has a very good prediction time (0.157 s) compare to the others in this study, although its requires a longer training time (155.513 s).

TABLE V
NAIVE BAYES CLASSIFIER

Class	Count	Recall	Precision	Specificity
0	51365	0.000000	0.000000	1.000000
1	2804	0.84700	0.987583	0.999557
2	3730	0.992761	0.066375	0.220934
3	533	0.964218	0.806299	0.998244
4	1142	0.785276	0.988962	0.999856
5	351	1.000000	1.000000	1.000000
6	368	0.000000	0.000000	0.997209
7	10294	1.000000	1.000000	1.000000
Training time		2.968 s		
Prediction time		0.143 s		

4) Naive Bayes: Naive Bayes neither detects normal instances (category 0) nor Denial-of-service attacks (category 6). It also has a very poor precision score on Complex Malicious Response Injection attacks (category 2) yielding to high False Positives. Naive Bayes however has a good training and prediction time (2.968 s and 0.143 s respectively).

Figure 3 clearly shows that in terms of training time, Naive Bayes (2.968 s) , followed by Decision Tree (7.841 s) outperform Random Forests and Multilayer Perceptron algorithms which 10.018 s, 24.731 s and 155.513 s respectively. It is however important to note that the Multilayer Perceptron is penalized by the required maximum iterations number (set to 100) for optimization purpose.

Fig. 3. Performance Analysis of different intrusion detection methods in terms of training time Time

In terms of prediction time, the Naive Bayes (0.143 s) still outperforms the other algorithms Multilayer Perceptron,Decision Tree and Random Forests at 0.227s, 0.234 s and 0.344 s respectively. The Multilayer Perceptron is almost as fast as the Naive Bayes for prediction. That means that it may be computer intensive to train and optimize, but very fast during prediction of new samples.

V. CONCLUSION AND FUTURE WORK

We have proposed a comparison of big data machine learning approaches by using an anomaly detection framework built with Apache Spark, for multi-class anomaly detection in a SCADA dataset. Decision Tree, Naive Bayes, Random Forests, Multilayer Perceptron are selected for the performances comparison using recall, precision, specificity, training time and prediction time criteria. The results show that the Decision Tree approach is very effective in the detection of several classes of attacks in the SCADA systems except the DoS attacks. It also has a good training and prediction time. Random Forests is also doing good in all classes of attacks (more that 95%) except DoS (60 %). But it has a longer training and detection time compare to Decision

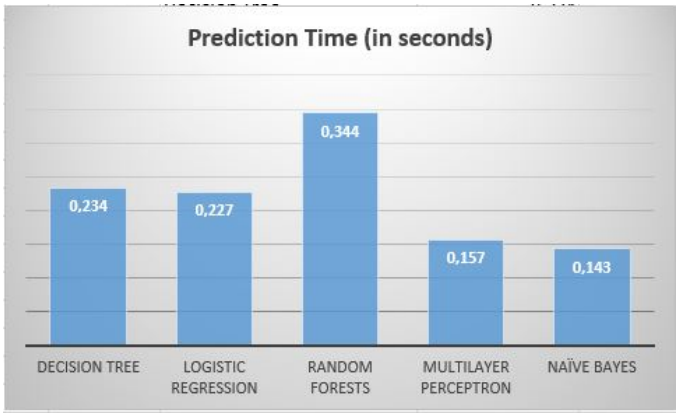

Fig. 4. Performance Analysis of different intrusion detection methods in terms of Prediction Time.

Trees. Nave Bayes and Multilayer Perceptron on the other hand give an overall poor classification results, but Nave Bayes is very fast at training and detecting (2.96 s and 0.14 s respectively) . Multilayer Perceptron, while taking time to train (155.51 s) is very fast in the prediction phase (0.16 s). We plan to undertake a future research to find out why the different classifiers are acting poorly in DoS attacks detection. For future contributions, the proposed framework could be extended to include data acquisition module using tools like sqoop, Logstash or Kafka. Sqoop would enable the use of the data historian database for training purpose to generate the models. With an architecture using Kafka and/or Logstash, we could define a time frame to collect the data streams from the various SCADA sensors and use the micro-batch capability of Spark streaming to perform near real-time anomaly detection in the SCADA networks. Future work could as well extend the comparison to classification algorithms as Support Vector Machines, Gradient-Boosted Tree, One-vs-Rest classifiers. Finally, labeled datasets like the one used in our experimentation are costly as they require an expert and they are rarely available in real life. Future research could investigate the usage of unsupervised or semi-supervised approaches for anomaly detection in SCADA systems with Apache Spark.

REFERENCES

[1] Keith Stouffer, Joe Falco, and Karen Scarfone, Guide to industrial control systems (ICS) security, in: NIST special publication 800.82 (2011), pp. 1616.

[2] Christophe Kolsky, Interfaces Homme-machine : application aux systmes industriels complexes, ed. by Herms, 1997.

[3] Stuart A Boyer, SCADA: supervisory control and data acquisition, International Society of Automation, 2009.

[4] Rafael Ramos Regis Barbosa, Anomaly Detection in SCADA Systems-A Network Based Approach, in: (2014).

[5] Paul Oman, Edmund Schweitzer, and Deborah Frincke, Concerns about intrusions into remotely accessible substation controllers and SCADA systems, in: Proceedings of the Twenty-Seventh Annual Western Protective Relay Conference, vol. 160, 2000.

[6] Alfonso Valdes and Steven Cheung, Intrusion monitoring in process control systems, in: System Sciences, 2009. HICSS09. 42nd Hawaii International Conference on, IEEE, 2009, pp. 17.

[7] Bonnie Zhu, Anthony Joseph, and Shankar Sastry, A taxonomy of cyber attacks on SCADA systems, in: Internet of things (iThings/CPSCom), 2011 international conference on and 4th international conference on cyber, physical and social computing, IEEE, 2011, pp. 380388.

[8] Jill Slay and Michael Miller, Lessons learned from the maroochy water breach, in: International Conference on Critical Infrastructure Protection, Springer, 2007, pp. 7382.

[9] Bill Miller and Dale Rowe, A survey SCADA of and critical infrastructure incidents, in: Proceedings of the 1st Annual conference on Research in information technology, ACM, 2012, pp. 5156.

[10] Nicolas Falliere, Liam O Murchu, and Eric Chien, W32. stuxnet dossier, in: White paper, Symantec Corp., Security Response 5 (2011), p. 6.

[11] Robert M Lee, Michael J Assante, and Tim Conway, Analysis of the cyber attack on the Ukrainian power grid, in: SANS Industrial Control Systems (2016).

[12] Evan Perez, Alleged Russian malware found on Vermont utilitys laptop, in: CNN (2017).

[13] Alvaro A Crdenas et al., Attacks against process control systems: risk assessment, detection, and response, in: Proceedings of the 6th ACM symposium on information, computer and communications security, ACM, 2011, pp. 355366.

[14] Bonnie Zhu and Shankar Sastry, SCADA-specific intrusion detection/prevention systems: a survey and taxonomy, in: Proceedings of the 1st Workshop on Secure Control Systems (SCS), vol. 11, 2010.

[15] Manish Kulariya, Priyanka Saraf, Raushan Ranjan and Govind P. Gupta, "Performance analysis of network intrusion detection schemes using Apache Spark". In: Communication and Signal Processing (ICCSP), 2016 International Conference on (pp. 1973-1977). IEEE.

[16] Mohiuddin Ahmed, Adnan Anwar, Abdun N. Mahmood and Zubair Shah and Michael J. Maher, "An investigation of performance analysis of anomaly detection techniques for big data in scada systems". In : EAI Endorsed Trans. Indust. Netw. and Intellig. Syst, 2015

[17] Justin M. Beaver, Raymond C. Borges-Hink and Mark A. Buckner, "An evaluation of machine learning methods to detect malicious SCADA communications". In : Machine Learning and Applications (ICMLA), 2013 12th International Conference on. IEEE, 2013. p. 54-59.

[18] Aleksandar Lazarevic, Levent Ertoz, Vipin Kumar, et al., "A comparative study of anomaly detection schemes in network intrusion detection". In : Proceedings of the 2003 SIAM International Conference on Data Mining. Society for Industrial and Applied Mathematics, 2003. p. 25-36.

[19] Mustapha Belouch, El Hadaj Salah and Idhammad Mohamed . "Performance evaluation of intrusion detection based on machine learning using Apache Spark.". In: Procedia Computer Science 127 (2018): 1-6.

[20] Nahla Ben Amor, Salem Benferhat and Zied Elouedi, Naive bayes vs decision trees in intrusion detection systems, in: Proceedings of the 2004 ACM symposium on Applied computing, ACM, 2004, pp. 420424.

[21] Leo Breiman, Random forests, in: Machine learning 45.1 (2001), pp. 532.

[22] Marko Robnik-ikonja, Improving random forests, in: European conference on machine learning, Springer, 2004, pp. 359370.

[23] Christian Leistner et al., Semi-supervised random forests, in: Computer Vision, 2009 IEEE 12th International Conference on, IEEE, 2009, pp. 506513.

[24] Thomas Morris et al., A control system testbed to validate critical infrastructure protection concepts, in: International Journal of Critical Infrastructure Protection 4.2 (2011), pp. 88103.

[25] Miroslav Kubat," An Introduction to Machine Learning" in; Springer, 2015.

[26] iawei Han, Micheline Kamber, and Jian Pei, "Data mining: Concepts and techniques" in: Elsevier, 2012.

[27] Chun-Wei Tsai et al., Big data analytics: a survey, in: Journal of Big Data 2.1 (2015), p. 21.

[28] Wei Gao et al., On SCADA control system command and response injection and intrusion detection, in: eCrime Researchers Summit (eCrime), 2010, IEEE, 2010, pp. 19.

[29] Thomas Morris and Wei Gao, Industrial control system traffic data sets for intrusion detection research, in: International Conference on Critical Infrastructure Protection, Springer, 2014, pp. 6578.