Performance/Cost Analysis of a Cloud Based Solution for Big Data Analytic: Application in Intrusion Detection

[1]Nada Chendeb Taher, [1]Imane Mallat, [2]Nazim Agoulmine, [3]Nour Mawass
[1]Lebanese University, Faculty of Engineering, Tripoli, Lebanon
[2]COSMO, IBISC Laboratory, UEVE, Paris-Saclay University, France
[3]Normandie University, UNIROUEN, LITIS

***Abstract*—The essential target of 'Big Data' technology is to provide new techniques and tools to assimilate and store large amount of generated data in a way to analyze and process it to get insights and predictions that can offer new opportunities towards the improvement of our life in different domains. In this context, 'Big Data' treats two essential issues: the real-time analysis issue introduced by the increasing velocity at which data is generated, and the long-term analysis issue introduced by the huge volume of stored data.**

To deal with these two issues, we propose in this paper a Cloud-based solution for big data analytic on Amazon Cloud operator. Our objective is to evaluate the performance of Big Data services offered regarding the volume/velocity of the processed data. The dataset we use contains information about "network connections" in approximately 5 million records with 41 features; the solution works as a network intrusion detector. It receives data records in real time from a raspberry pi node and predicts if the connection is bad (malicious intrusion or attack) or good (normal connection). The prediction model was made using a logistic regression network. We evaluate the cloud resources needed to train the machine learning model (batch processing), and to predict the new streaming data with the trained network in real time (real time processing).

The solution worked very well with high accuracy and the results show that when working with Big Data in the cloud, we are mainly dealing with a cost/performance trade-off, the processing performance in term of response time for both long-term and real-time analysis can be always guaranteed once the cloud resources are well provisioned according to the needs.

I. Introduction

From the down of civilization until 2003, humankind generated five exabytes of data. Now we produce five exabytes every two days and the pace is accelerating [1]. Our smart phone collects data on how we use it and our web browser collects information on what we are searching for. Servers also collect information about connections and user activities to protect and develop the services they are offering. Today, it is hard to imagine any activity or device that does not generate data. Just think about all the pictures we take on our smart phones. We upload and share 100s of thousands of them on social media sites every second. With the datafication of everything, comes Big Data.

Cloud Computing and Big Data are distinct disciplines that have evolved separately over time. However, they are increasingly becoming interdependent. The concept of Big Data has been around for many years, but its mainstream application started only recently [2]. The concept of cloud computing also traces back to the 1960s and has since then evolved and passed through many stages to become today a mainstream commercial necessity [2].

Demand for Big Data is calling for the adoption of Cloud platforms because Big Data techniques need very high computing resources that grow in parallel with the fast growth of the generated data. Therefore, if we need to transform the data into value and utilize its potential, we need first to fully embrace Cloud-based systems.The convergence of Big Data and Cloud Computing eventually provides new opportunities and applications in many verticals. Nowadays, many companies have already adopted Cloud technologies as a mean to store and analyze data, and finally get predictions and insights about their business performances. Many examples of these applications do exist in many domains such as healthcare, industry 2.0, Networking, business, marketing and many others.

Big data technology is also helping to fight hackers. Indeed, the same tools that are central to Big data can also used to analyze network traffic at large scale and react faster to attacks and therefore prevent damages or data leakage before they happen. Big Data can be used to identify anomalies in device behaviour, user behaviour or network connections.

In this context, any application with Big Data analytic can take one or both of two ways: the (1) real-time analysis that helps to find out irregularities in the collected data and act as fast as possible to prevent an undesired scenario, or the (2) long-term analysis that uses the massive data collected and utilizes the insights to identify the future trends and opportunities.

With the real-time analysis, we face the "Velocity" challenge. In major applications, we have to process data and get information and decision in real time before the next data is generated. For example, in network security, we have to detect any threat and stop it as early as the network connection starts to be established, avoiding individuals entering, interacting with or damaging the system. While with the long-term analysis that requires batch processing, we face the "Volume" challenge; where to store the huge amount of data and how

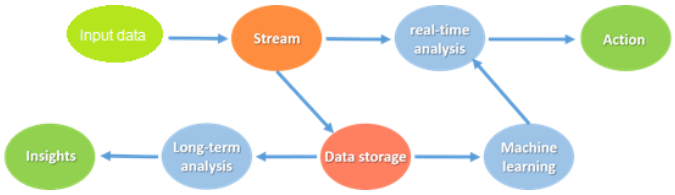

Fig. 1: A diagram for a Cloud based model for real-time and long-term analysis of Big Data

to process it accurately and in a reasonable amount of time and cost?
The two different ways of processing data lead to the necessity to use a Cloud based solution for Big Data analytic that is able to address both the volume and the velocity challenges as highlighted in Figure 1.

This model could be built on existing Cloud Computing infrastructure however these infrastructures (computing resources) do not offer the same level of Quality of Service neither in general the same charging model. Therefore, it is important to be able to evaluate the best offers in terms of performance and services. It is also important to identify what is the trade off between the provided performances and the cost. In particular, when the volume/complexity/velocity of the data increases, it is important to understand how the processing performance is positively or negatively impacted by the provided computing resources. These questions constitute the challenges we are addressing in this paper to derive a performance/cost model to execute Big Data analytic in the Cloud Computing for intrusion detection applications.

The data is related to "network connections" and the aim is to specify and deploy a machine learning solution able to detect malicious connections and stop them as soon as possible. Information about new data connection to the protected system is sent to intrusion detection system that is running in the Cloud and are processed in real-time (streaming). If the intrusion-detection system detects an anomaly, it blocks the connection and/or notifies the network administrator. The decision-making are the result of batch processing/machine learning applied to the data accumulated over time.
The remaining of this paper is as follows: after preliminaries presenting Big Data and Big Data analysis in section II, the following section III presents the proposed intrusion detection solution based on Cloud Computing and Big Data analytic. The solutions uses the services provided by Amazon Cloud provider namely AWS (Amazon Web Services). The section IV, presents an implementation of the system as well as the used dataset and the machine learning model, in this section we present also the results of the conducted performance/cost analysis. In section V, we discuss the results and highlight the cost-benefit of our solution against a traditional one. Finally, we conclude the work in section VI and present some perspectives.

Fig. 2: Big Data 4V's

II. Preliminaries

A. What is Big Data?

There is no stable definition for Big Data. We can use Big Data to describe a massive volume of both structured and unstructured data that is so large, increasing very fast and that it is very difficult to process using traditional tools, databases and software techniques. To deal with this kind of data, a multitude of tools and frameworks are available including the famous Hadoop (and spark) ecosystem. The underlying techniques behind these tools and frameworks are distribution, parallelism and clustering of computing resources.

B. Characteristics of big data

Big data is commonly characterized using a number of V's. The first four are Volume, Velocity, Variety and Veracity as shown in Figure 2. Volume refers to the vast amount of data that is generated every second, minutes, hour, and day in our digitized world. Variety refers to the ever increasing various forms of captured data such as text, images, voice, geospatial, raw, etc. Velocity refers to the speed at which data is being generated and the pace at which data moves from one point to the next. Veracity refers to the quality and the reliability of the data.

C. Turning Data into Value

Data analytic backed by the expansion of computing power is enabling companies to extract maximum value from data to get the best insights. The way they help us move from data to insight and value is called the 'wisdom hierarchy' [3]. The wisdom hierarchy is a conceptual framework for thinking about how the raw inputs of reality (signals) are stored as data and transformed first into information, then into knowledge, and finally into wisdom (Figure 3). It summarizes the data analysis process. In other words, 'Wisdom Hierarchy' represents a path from gathering and exploring raw data, to machine learning that enables getting knowledge from raw data, and finally to artificial intelligence that ensures a

Fig. 3: Wisdom hierarchy

deep understanding of knowledge. Based on this hierarchy, we discover the importance of machine learning in big data domain to get insights from raw data, make predictions and then take the appropriate decisions; this is what we call the 'data mining'. So what is machine learning? And what are the machine learning models?

D. Machine learning

Machine learning is an artificial intelligence technique that allows software applications to become more accurate in predicting outcomes without being explicitly programmed. The basic premise of machine learning is to build algorithms that can receive a large amount of input data and then predict a reliable output. Based on this definition, Machine Learning takes data as input. The input data is called 'training data'. The desired output is called 'Targets' or 'Labels'. Machine learning models are often categorized as supervised or unsupervised. Supervised models need humans to provide both input ('Training Data') and the desired output ('Targets'). Once training is complete, the algorithm must be tested on new labeled data to compute its 'accuracy'. The accuracy parameter represents the number of correct predictions from all predictions made with the Machine Learning algorithm. The desired accuracy depends on the application we deal with. As for the unsupervised models, the training data does not include 'Targets' so we do not tell the system where to go, the system has to understand itself from the data provided.

E. Data analysis process

When talking about 'Wisdom Hierarchy', data analysis passes through many steps before being translated into actions. In the Big Data course provided by UCSD untitled 'Big Data Specialization' [4], these steps are clearly defined and detailed. They start with data acquisition towards decision-making.

1) Acquiring data: The first step in acquiring data is to determine what data is important. Leaving out even a small amount of important data can lead to incorrect conclusions.

2) Exploring data: Exploring data is a part of the two-step data preparation process. We need to do some preliminary investigation in order to gain a better understanding of the specific characteristics of the data. In this step, we will be looking to things such as correlations, general trends, and outliers.

3) Pre-Processing data: There are two main goals in the data pre-processing step. The first is to clean the data to address data quality issues, and the second is to transform the raw data to make it suitable for analysis.

4) Analyzing data: Data analysis involves building a model from the clean data, which is called input data. The input data is used by the analysis technique to build a model. What the model generates is the output data. There are different types of problems, and so there are different types of analysis techniques. The main categories of analysis techniques are classification, regression, clustering, association analysis, and graph analysis.
In classification, the goal is to predict the category of the input data. When the model has to predict a numeric value instead of a category, then the task becomes a regression problem. In clustering, the goal is to organize similar items into groups. The goal in association analysis is to come up with a set of rules to capture associations between items or events.
Let's briefly look at how to evaluate each technique. For classification and regression, we will have the correct output for each sample in the input data. Comparing the correct output and the output predicted by the model provides a way to evaluate the model. For clustering, the groups resulting from clustering should be examined to see if they make sense for our application. For association analysis, some investigation will be needed to see if the results are correct.
As a summary, data analysis involves selecting the appropriate technique for our problem, building the model, and then evaluating the results.

5) Turning insights into actions: The next step is to determine what action or actions should be taken, based on the insights gained? This is the first step in turning insights into action. Now that we have determined what action to take, the next step is to study how to implement the action.

F. Techniques and tools for big data analysis

Hadoop is an open-source framework that allows to store and process large datasets in parallel and distributed fashion. It runs on clusters of commodity servers and can scale up to support thousands of hardware nodes and massive amounts of data. Consequently, Hadoop became a data management platform for big data analytics. As the diagram of Figure 4 shows, we have different layers that can operate in this ecosystem:

- The Hadoop 'HDFS' as Hadoop Distributed File System, that is a parallel and distributed storage unit.

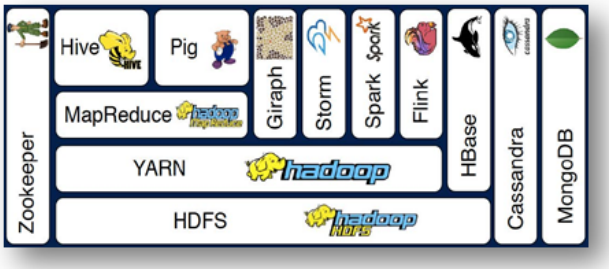

Fig. 4: Layers diagram in Hadoop ecosystem

- The Hadoop 'Yarn' as Yet Another Resource Negotiator, a resource manager layer that interacts with application and schedules resources for their use
- 'MapReduce' that is a programming model for processing large amounts of data in parallel and distributed fashion composed of Map() and Reduce() procedures
- 'Storm' platform, a distributed, real-time data processing platform
- 'Spark' platform, an open-source big data processing framework built around speed, ease of use, and sophistical analytics. In addition to MapReduce operations, it supports SQL queries, streaming data, machine learning and graph processing. It also offers a shell for python.
- 'Cassandra', 'MongoDB', 'HBase' are distributed databases

In our cloud-based solution, we will use the 'Spark' framework above a Hadoop cluster so that we can ensure a real-time processing and sophistical analytics like 'Machine Learning' and 'data mining'.

G. Cloud Computing

Cloud Computing is a paradigm in which any user with internet connection can rent computing resources as needed from a cloud operator owning large datacenters and offering services. This is mainly an economic revolution in the IT/Networking field thanks to the huge advances in virtualization technology and datacenters. Cloud Computing services cover a vast range of options going from the basics of storage, networking, and processing power through natural language processing and artificial intelligence services. A fundamental concept behind cloud computing is that the location of the service, and many of the details such as the hardware or many benefit operating system on which it is running, are largely irrelevant to the user. There ares from cloud computing that make this field a very important one. Three of the main benefits of cloud computing are self-service provisioning, elasticity and Pay per use.

H. Literature review

With the exponential growth of data and the digitization of everything, cyber-attacks become widespread and threatens the organization's security and the personal privacy. In the other hand, with the evolution of big data technology, new frameworks and techniques appeared providing solutions for real-time and complex analysis towards trends and behavior discovering, malicious preventing, and fraud detections. Big Data analytics promises significant opportunities for solving different information security problems [5]. For this reason, researches and works evolved in the domain of Big data and cyber-security.

Security is now a big data problem because the data that has security context is huge. Hence, to construct a big data processing and computing infrastructure extra secure, authors in [6] summarized some security and privacy related issues.

Under a big data environment, it is more complicated and difficult to store and process the organizations and the customer's information in a secure manner given the huge volume, the increasing velocity and the variety of generated data. In this context, the author of [7] proposed some solutions for intrusion detection and threats attacks limitation. One of these solution was to implement a MapReduce machine learning model that can distinguish between bad and normal connections based on some features and metrics such as flow duration, average bytes per packet in the flow, and average bytes per second in the flow. The proposed model may use the collected network traffic consisting of both normal flow and potential attack flows, to train a logistic regression (LR) or naïve Bayes network that works as binary classifier.

One of the machine learning techniques developed under the big data and cyber security domains is the neural network approaches that takes an interest role for discovering patterns and malicious activity of the users. Ana-Maria Ghimeș and Victor-Valeriu Patriciu, proposed a neural network model that consists of several case studies on algorithms and architectures of neural networks for determining the best way to discover new attacks malicious patterns in data [8].For test purposes, they used data sets provided by UCI Machine Learning Repository [9]. For classification, they have used repositories like "Detect Malicious Executable (AntiVirus) Data Set" which consists of malicious and non-malicious samples. They have started the study with a simple implementation of a neural network and continued using pruning techniques for finding the optimal network. The best model they have obtained in the pruning process had the hyperbolic tangent function as the activation function, and consisted of 4 layers: the input layer with 22 neurons, an output layer with one neuron and two hidden layers with 8, respectively 5, hidden neurons. Once the pruning will be completed the training process will be initiated for updating the connection weights and getting the best performance.

III. Proposed Cloud Based Solution for Intrusion Detection

As the speed and the volume of network data increases in particular connections to remote servers, the need to perform the data analysis in real time with machine learning algorithms and extract a deeper understanding from the data becomes

crucial for all business, organizations and governments. As the same time, to satisfy the increasing velocity and volume even the complexity of generated data, the use of big data tools and techniques that ensure parallelism in computing, scalability and reliability is a necessity.

Consequently, and based on the importance given for both the real time and the long-term analysis in the majority of the domains today, we created a cloud based system for both stream and batch processing using the 'Amazon' Cloud that provides all Big Data techniques and tools we need to perform such a system, and at a low cost without the necessity to procure hardware or to maintain infrastructure. To perform our solution on Amazon we referred to a set of Amazon Web Services (AWS) that can be connected between them. Some of these services are for capturing data streams, other for compute and processing and some other for storage and notification. In the following we will describe the services we used in our model.

A. Used AWS services

1) AWS IoT core: Amazon IoT service is used to connect IoT devices, receive data from these devices using 'MQTT' protocol and publish the messages to a specific 'topic'. In IoT AWS, a 'thing' represents any connected device. Additionally, AWS IoT 'rules' applied on the received data, gives IoT-enabled devices the ability to interact with other AWS services. One rule can combine more than one 'actions'.

This service costs as today 0.08$ per million minutes of connection, 1$ per million messages and 0.15$ per million rules triggered.

2) Amazon Kinesis Stream: Amazon Kinesis Streams can continuously capture and store terabytes of data per hour and hundreds and thousands of sources. Data records are accessible for a default of 24 hours from the time they are added to a stream. During that window, data is available to be read, re-read, backfilled and analyzed, or moved to long-term storage. With amazon kinesis streaming data can be ingested, buffered and processed in real-time, so insights can be derived in seconds or minutes instead of hours or days.

Pricing is based on two core dimensions - Shard Hour and PUT Payload Unit. 'Shard' is the base throughput unit of an Amazon Kinesis stream. An Amazon Kinesis stream is made up of one or more shards. Each shard provides a capacity of 1MB/sec data input and 2MB/sec data output. Each shard can support up to 1000 write and 5 read transactions per second. The number of shards needed within the stream is specified based on the throughput requirements. The charging of the shard is based on hourly usage rate.

A record is the data that the producer adds to the Amazon Kinesis stream. A PUT Payload Unit is counted in 25KB payload "chunks" that comprise a record. For example, a 5KB record contains one PUT Payload Unit, a 45KB record contains two PUT Payload Units, and a 1MB record contains 40 PUT Payload Units. PUT Payload Unit is charged with a per million PUT Payload Units rate.

For each Shard, the cost is 0.015$ per hour, and 0.014$ per million PUT Payloads Units.

3) Amazon Elastic Compute Cloud (EC2): Amazon EC2 provides scalable computing capacity in the Amazon Web Services (AWS) cloud. Using Amazon EC2 eliminates the need to invest in hardware up front allowing to develop and deploy applications faster. Amazon EC2 can be used to launch as many or as few virtual servers as needed, configure security and networking, and manage storage. With On-Demand instances, only EC2 instances usage is charged on per hour depending on which EC2 instance type is used. For example, for c5.2xlarge instance type (8 CPUs and a RAM of 16GiB), the cost is 0.34$ per hour.

4) Elastic MapReduce (EMR): Amazon EMR is a highly distributed computing framework to easily process and store data quickly in a cost-effective manner. Amazon EMR uses 'Apache Hadoop', an open source framework, to distribute the data and processing across a resizable cluster of Amazon EC2 instances and allows to use the most common Hadoop tools such as 'Hive', 'Pig', 'Spark' and so on. With Amazon EMR, more core nodes can be added at any time to increase the processing power.

Amazon EMR pricing is simple and predictable: we pay a per-second rate for every second we use, with a one-minute minimum. For example, a 10-node cluster running for 10 hours costs the same as a 100-node cluster running for 1 hour. The hourly rate depends on the instance type used. For example, for c5.2xlarge EC2 instance type the cost is 0.085$ per hour.

5) Simple Storage Service (S3): Amazon S3 is carefully engineered to meet the requirements for scalability, reliability, speed, low-cost, and simplicity.

We pay 0.023$ per GB for the first 5 TB per month, 0.022$ per GB for the next 450 TB per month, and 0.021$ per GB for over 500 TB per month.

6) Short Notification Service (SNS): SNS is a fully managed push notification service that allows sending individual messages to large numbers of recipients. Amazon SNS makes it simple and cost-effective to send push notifications to mobile device users, email recipients or even send messages to other distributed services.

SMS messages sent to non-US phone numbers are charged. For example, to send a message to Lebanon, the cost per message is 0.04746$ for Alfa line and 0.05192$ for Touch line.

B. Our proposed model

These services should be carefully connected to form our Cloud based solution for real-time and batch processing of Big Data as in Figure 5.

First, we have created a spark cluster of specific EC2 instance type using Amazon EMR. After uploading data to an S3 bucket, data is pulled from spark cluster to train a machine learning network. A Raspberry Pi (Rpi) that plays the role of any connected device is connected to the Amazon IoT core. Data sent from the Rpi is published to a specific topic where a kinesis rule is applied to push data into the kinesis stream

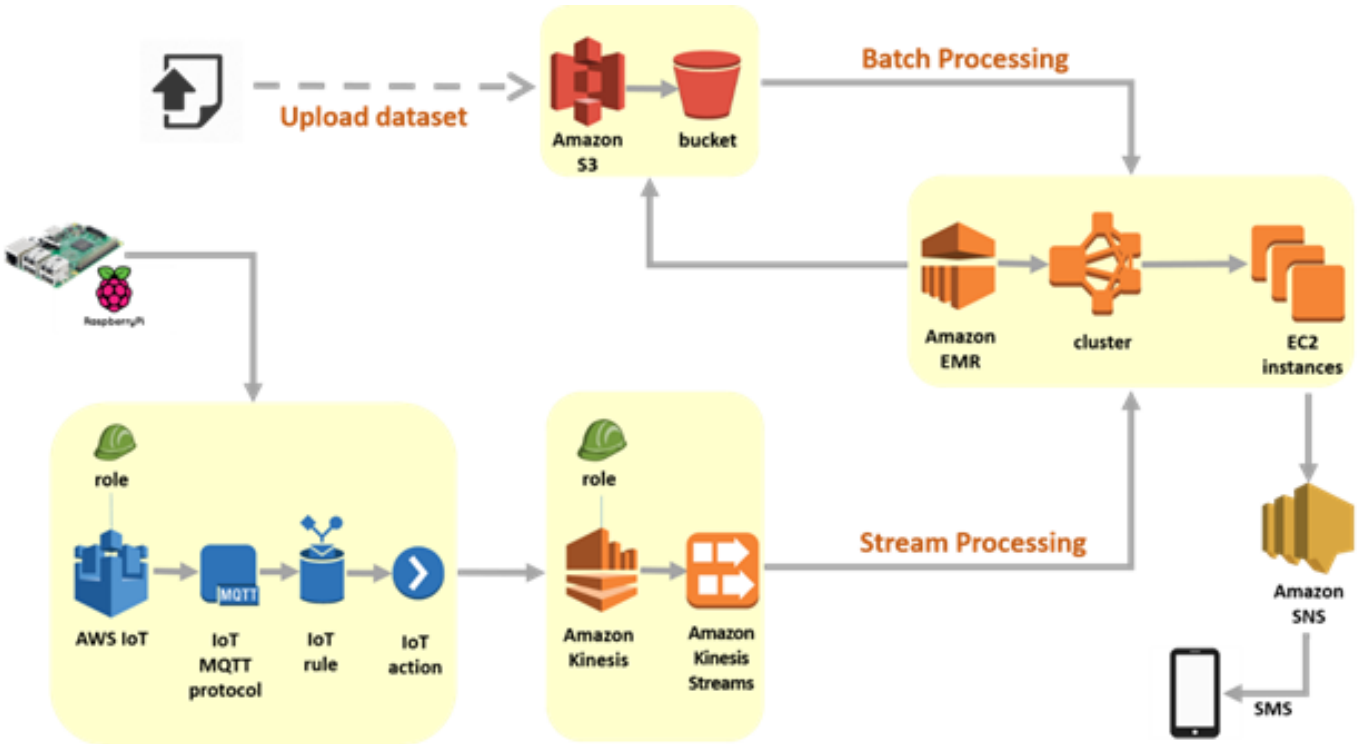

Fig. 5: Our Cloud Based solution for Big Data analytic

shard. In the EMR spark cluster, streaming data is pulled to be predicted with the machine learning model already built. In case of anomalies, a notification is sent with Amazon SNS to some specific phone number.

IV. DataSet, Processing Mechanism and Results

A. Network connections DataSet

To build this solution, we have used the 'Kdd Cup' dataset. This is a typical big dataset that helped us to perform the performance/cost analysis of the complete model in the cloud. This dataset is the one used for The Third International Knowledge Discovery and Data Mining Tools Competition. The competition was to evaluate different network intrusion detectors i.e. predictive models capable of distinguishing between 'bad' connections, called intrusions or attacks, and 'good' normal connections.

This dataset contains records about network connections with 41 features, like connection duration, the number of segments sent, the number of failed segments, the protocol (tcp or udp), the service (http, ...) used, the label (in case of labeled data) that indicates the status of the connection ('normal', 'buffer-overflow', 'guess-passwd.', ...), etc...

This dataset consists of 6 compressed data files. Each file contains distinct set of data including what is labeled to use for training and testing the network and what is unlabeled to use for prediction [10]. In our experiment, we have used the full data file 'kddcup.data.gz' for training network, the 'corrected.gz' data file with corrected labels for testing the machine learning network and the unlabeled data file 'kddcup.testdata.unlabeled-10-percent.gz' to stream data from the Rpi thus simulating new connections that should be analyzed in real-time. Figure 6 shows a portion of the full data file 'kddcup.data.gz'.

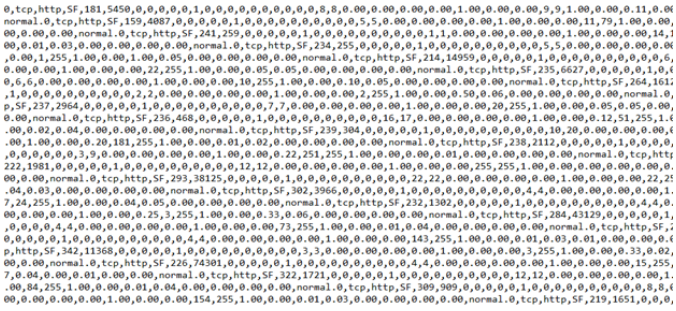

Fig. 6: Sketch of a portion of the data used

B. Processing mechanism

We have used the 'logistic regression', a supervised machine learning model, on this dataset to distinguish between the normal and the bad network connections. The Machine learning model was inspired from a tutorial applied on the same data set testing different algorithms in Apache Spark [11]. The following is a quick walk-through of the Process that is presented in Figure 5:

- First, we uploaded the full dataset 'kddcup.data.gz' with the 'corrected.gz' dataset to a S3 bucket.

	2M records	3M records	5M records
3 instances	—	—	—
5 instances	152.58	—	—
6 instances	142.3	—	—

TABLE I: Training time (in seconds) for different c5.xlarge cluster sizes and different volumes of training data.

	2M records	3M records	5M records
3 instances	118.17	162.496	—
5 instances	101.22	132.9	—
6 instances	90.4	129	—

TABLE II: Training time (in seconds) for different c5.2xlarge cluster sizes and different volumes of training data.

- We have created a spark cluster on EMR to process the large amount of data. In this solution we used a cluster of 3 EC2 instances with c5.2xlarge instance type (8 CPU and 16GB RAM for each instance).
- On EMR cluster, we have created a machine learning "Logistic Regression" model, that takes as 'Input' the numerical features from 'kddcup' data and as a 'Target' the label '0' in case of a normal connection and '1' in case of a bad connection. We have used 2 Million records of 'kddcup.data.gz' for training the network. The 'corrected.gz' data file is also used to compute the accuracy and to validate this network.
- We downloaded the 'kddcup.testdata.unlabeled-10-percent.gz'data file on the raspberry pi. We have pulled records from this file and sent them to the Amazon IoT core. The generated data is published to a topic on which a rule is applied to send data to a kinesis stream. Then, on the EMR cluster, we have pulled each record from the Kinesis stream to predict it using the trained machine learning network.
- In case of a bad record (bad network connection), a notification is sent to a specific phone number using Amazon SNS service.

C. Performance/Cost Analysis of the Training Phase

To perform this machine learning, we have tried different types of EC2 instance as well different numbers of instances in the cluster. The objective is to derive the best cost-effective configuration (i.e. a trade-off between the processing time and the cost). After several configuration tests in the real Amazon cloud, we have been able to complete Table 1 and Table 2.

As shown in Table 2, we notice that a c5.2xlarge instance type cluster with 3 instances can train network a max of nearly 3M records. With the same type of virtual machine but with 5 instances, it is possible to reach a maximum of 3.5M records. Eventually, when we increased the number of instances to 6, we became able to train the network with all the available records in the 'kddcup.data.gz' data file i.e. 5M records.

When using a c5.xlarge cluster with 5 instances or even with 6 instances, it was not possible to train the network with more than 2M records nearly as shown in Table 1. Hence, we noticed that the processing time is decreasing with the decreasing size of the cluster as well as with increasing type of instance. Since the training with 2 Million records has given us a good accuracy equal to 0.9195, we found it the right configuration to train the network.

From a cost perspective, using a c5.2xlarge cluster, the usage cost for each EC2 instance is 0.34$/h, and 0.085$/h for EMR. For a c5.xlarge cluster, cost decreases to 0.199$/h for each EC2 instance, and 0.052$/h for EMR. If we had used a c5.xlarge cluster with 5 instances, the total cost per hour would have been the following: 5*0,199+0.052=1,047$/h and the training would have taken 152.581 secs. Similarly, if we had used c5.2xlarge cluster with 3 instances, the total cost per hour would have been: 3*0.34+0.085=1.105$/h and the training would have take 118.17 secs. Therefore, we have decided to use a cluster of 3 c5.2xlarge instances to train the network with 2M records (480MB) .

Regarding the response time of the analysis we wanted to be as real-time as possible, we noticed that all records did not exceed 150 Bytes. It was therefore enough to use only one Shard, on condition that the time between two sent records did exceed the 1 ms, while the time between two received records will be in minimum 0.2 sec. The obtained response time was very low and did not eventually exceed the ms. If we had to send or receive faster or bigger records, we would have needed to increase the number of Shards. Kinesis Stream is a very efficient service to ensure the scalability of our model as well as maintaining the stream processing real-time. For each created shard, the price was 0.015$/h while it was 0.014$ for each million PUT Payloads. In our case, each record contained only one PUT Payload. In one second, we generated 1000 records and 3600000 Put Payloads per hour. Therefore, the bill to pay was 4*0.014+0.015=0.071$/h

V. Performance/Cost Analysis of the Intrusion Detection Phase

A. Presentation of the Case Study

Assume we have 20 computing devices that all generate the same type of data (the one used to build our model). Assume that each record does not exceed 150 Bytes, therefore each record can be eventually included in one PUT Payload Unit. Assume also that each device generates data at a speed of 50 records per second, therefore one shard is required to ensure this input velocity (each shard will actually support 1000 write transactions per second corresponding to the 20 devices generating 50 writes per second). Assume the required output velocity is at least 1 output record per second. To achieve this performance, it is necessary to subscribe to 4 shards (since each shard supports 5 read transactions per second for a total

of 4*5=20 read transactions per second for the 4 shards).
The output velocity requirement depends on the consuming application. For example, some applications may need to process data very quickly (critical application) while other applications may accept process the date more slowly.
Suppose, a spark cluster is built on Amazon EMR with 3 c5.2xlarge instances to read output records from shards and process them in real-time.
Assume the used prediction model can detect 70 anomalies in average per month therefore 100 messages will be sent monthly.

B. Monthly Cost Analysis

In this section, we will evaluate the monthly cost for the case we assumed above. If the used configuration is the c5.2xlarge instance type, the cost will be 0.34$/h for each instance and 0.085$/h for EMR. Therefore, the cost for the whole Spark cluster will be 0.34*3+0.085=1.105$ per hour. We can deduce the monthly cost, that is in this case 1.105$*24*30=795.6$.
To ensure a real time processing, the data records will be streamed using the Amazon kinesis. For each shard, the cost will be 0.015$/h and for each million of PUT Payload Unit, the cost will be 0.014$. Assuming 1000 records are generated per second and 4 shards are used for the streaming, the monthly cost will be 0.015*24*30=10.8$ for all the used shards, and (0.014*3600*24*30*1000)/1000000=36.288$ per month for the PUT Payload Units. We can deduce the cost for the amazon kinesis stream service usage that is 36.288+10.8=47.088$ monthly in our case.
For the Alfa line messaging service usage, the cost per sent message will be 0.04746$. If we suppose that we need to send a mean of 70 messages per month, the monthly usage cost for this service will be 70*0.04746=3.3222$.
Assuming we decide to not exceed the free tier offered for the AWS IoT core and the Amazon S3, such a solution will cost in total nearly 835.2102 $ per month corresponding to the sum of the individual costs 795.6$+36.288$+3.3222$. This is obviously an advantageous solution compareed to hiring one engineer in the company.

C. Comparison with on premises solution

If the decision is deploy this model on premise, it is necessary to use a minimum of three powerful computers and pay for CapEX (hardware cost) and OpEX (engineers cost to configure the specific environment and software on hardware). One should not neglect maintenance cost of the used hardware and software. Hence, this solution doesn't ensure scalability since it is necessary to buy more hardware in case increase in the computing or storage resources demand. With Big Data services deployed in the Cloud Computing, it is possible to use hardware and software as commodities with guarantee of performance. In this case, there is no need to worry about installation, maintenance or upgrade since it is part of the Cloud Computing service.
For this reason, we plebiscite building this solution in the Cloud taking benefit of all services provided by the Cloud operators however doing an appropriate Performance/Costs analysis of the different possibilities of deployment of the service as the one presented in this paper.

This approach can also be applied to other applications with different response time or batch processing requirements, In case of machine learning, it is also to include in the study a cost/performance analysis of the network resources to deal with the size and the speed of the data to manipulate.

VI. Conclusions

In conclusion, the big data cloud based model we built can reach the desired results in terms of response time and accuracy, with a low cost relatively. We always have a cost/performance trade-off; the increase in complexity, speed, and volume of data leads to using more Cloud resources with higher features and hence paying more. So we have to define the exact needs on the cloud to reduce costs and then make an efficient model.
Using the Amazon Cloud that provides many services to address Big Bata analytic requirements, we built a cloud based solution that deals with the real-time and the batch processing issues. This complete solution can help meet the stringent business requirements in the most-optimized, performant, and resilient possible way. It can also be used in many domains that require real-time anomalies detection or a long-term analysis to get insights and trends from stored data. The role of the engineer is to provision the requested resources from the cloud operator to satisfy the needs with the low cost possible. Performance on the cloud is always guaranteed once the rented resources are sufficient for the needed processing use case.

Acknowledgments

This research was supported by The Lebanese University and CNRS Lebanon. Part of this work was also conducted in the frame of the PHC CEDRE Project N37319SK.

References

[1] Sampriti, Sarkar, "Convergence of Big Data, IoT and Cloud Computing for Better Future", *Analytics Insight*, 2017
[2] Eric Schmidt
[3] Wood, Adam Michael, "The wisdom hierarchy: From signals to artificial intelligence and beyond",O'Reilly Data Newsletter, 2017
[4] "Big Data Specialization. s.l.", University of California San Diego, 2018
[5] Rasim Alguliyev, Yadigar Imamverdiyev, "Big Data: Big Promises for Information Security", 2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT), 2014
[6] Aditya Dev Mishra, Yooddha Beer Singh, "Big Data Analytics for Security and Privacy challenges", International Conference on Computing, Communication and Automation (ICCCA2016), 2016
[7] M. S. Al-kahtani, "Security and Privacy in Big Data", International Journal of Computer Engineering and Information Technology, 2017
[8] Ana-Maria Ghimeș, Victor-Valeriu Patriciu, "Neural Network Models in Big Data Analytics", 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI), 2017
[9] *Machine learning repository* [Online]. Available: http://archive.ics.uci.edu/ml/
[10] *Knowledge Discovery and Data Mining Tools Competition 1999 Data* [Online]. Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99
[11] *Spark Python Notebooks* [Online]. Available: https://github.com/jadianes/spark-py-notebooks/blob/master/README.md