New Anomaly Network Intrusion Detection System in Cloud Environment Based on Optimized Back Propagation Neural Network Using Improved Genetic Algorithm

Zouhair Chiba *, Noreddine Abghour, Khalid Moussaid, Amina El omri and Mohamed Rida

LIMSAD Labs, Faculty of Sciences, Hassan II University of Casablanca, 20100, Casablanca, Morocco

Abstract: Cloud computing is distributed architecture, providing computing facilities and storage resource as a service over an open environment (Internet), this lead to different matters related to the security and privacy in cloud computing. Thus, defending network accessible Cloud resources and services from various threats and attacks is of great concern. To address this issue, it is essential to create an efficient and effective Network Intrusion System (NIDS) to detect both outsider and insider intruders with high detection precision in the cloud environment. NIDS has become popular as an important component of the network security infrastructure, which detects malicious activities by monitoring network traffic. In this work, we propose to optimize a very popular soft computing tool widely used for intrusion detection namely, Back Propagation Neural Network (BPNN) using an Improved Genetic Algorithm (IGA). Genetic Algorithm (GA) is improved through optimization strategies, namely Parallel Processing and Fitness Value Hashing, which reduce execution time, convergence time and save processing power. Since, Learning rate and Momentum term are among the most relevant parameters that impact the performance of BPNN classifier, we have employed IGA to find the optimal or near-optimal values of these two parameters which ensure high detection rate, high accuracy and low false alarm rate. The CloudSim simulator 4.0 and DARPA's KDD cup datasets 1999 are used for simulation. From the detailed performance analysis, it is clear that the proposed system called "ANIDS BPNN-IGA" (Anomaly NIDS based on BPNN and IGA) outperforms several state-of-art methods and it is more suitable for network anomaly detection.

Keywords: Cloud computing, Network intrusion detection system, Anomaly detection, Back propagation neural network, Optimization, Genetic algorithm.

1. Introduction

Nowadays, one of the fastest growing and most used technologies in the IT field is Cloud computing [1]. Cloud computing (CC) refers to an information technology (IT) paradigm that delivers convenient, on-demand ubiquitous network access to shared pools of configurable computing resources (e.g., networks, servers, storages, and applications) as "service" via the internet for satisfying computing demands of users [2]. With virtualization technology, cloud computing integrates massive computing, storage and network resources into a unified pool of resources, and provides them to users as service. That a large number of users sharing resources improves the utilization of IT resources, brings cloud computing platform with low-cost, convenience, scalability, high reliability and also reduces the average cost of using IT resources [3]. In addition, Cloud computing has allowed its users to pay for only the services which they have used or consumed that is called as "Pay-as-you-go", similar to a utility . The National Institute of Standards and Technology introduces cloud computing by considering its main features [4] (i.e., bandwidth, rapid flexibility, measurable, on-demand service and Resource Pooling) and its three services delivering models (i.e., software as a service (SaaS), platform as a service (PaaS) and infrastructure as a service (IaaS)). IaaS comprises of highly automated and scalable computer resources accompanied with storage in a Cloud and network empathy which can be available on demand. PaaS provides a rostrum on which specific software can be deployed or evolved. SaaS is much similar to the technique of software furnishing used erstwhile [5]. Other benefits of CC are hardware cost reduction (since customers of CC do not need to accommodate powerful processors or any hardware resources), automatic and fast upgrading/updating of services, tremendous capacity of storage, ubiquitous access to documents (users could access their required documents and applications just by connecting to the Internet wherever they are), parallel processing, acceleration and time saving [6]. Thanks to these advantages, cloud computing technology has been widely adopted and used in commercial sector. Number and scale of public and private clouds are in high speed growth. With these characteristics, CC provides a great number of businesses and individuals with large development opportunities [7].
The architecture of CC characterized by distributed technology and big data applies such technology as virtualization and multi-tenancy, which bring vulnerabilities, security and sharing risks specific to cloud computing. Also, uploading sensitive data to public cloud storage services poses security risks such as accessibility, confidentiality and integrity to organizations that used CC. Moreover, non-stop cloud services have caused high levels of abuse and intrusion [3, 8]. Intrusions are actions that threaten the integrity, authenticity of information in a network resource by circumventing the measures provided for security [9].Further, intrusions are defined by NIST (National Institute of Standard and Technology) as attempts to endanger security policies (privacy, integrity and availability) or skip computer and network security mechanisms [6]. In fact, in CC there are two concerns of security issues; the first one regardless to cloud service provider which has guarantee that the services provided and cloud infrastructure are safe and secure, the second to the consumers including guarantee that the information and data of consumers are protected

[10,11].Therefore, to overcome these issues, intrusion detection system(IDS) besides firewall are used to protect resources and data in the cloud environment from attacks and malicious activities, while maintaining performance and service quality. Firewall cannot be used to detect insider attack, and some of attacks, like denial of service (DoS) and distributed denial of service (DDoS) are too complex for firewalls. So, it is the inevitable to a developed intrusion detection system in a cloud environment [8, 10]. Intrusion detection system is the process of identifying various events occurring in a system/network and analyzing them for the possible presence of intrusion. Then, an alarm will be raised and sent to computer network manager if there is probably intrusion [12]. In addition, NIST considers intrusion detection systems as software/hardware systems to automate the intrusion detection procedure [13]. Since the network is the backbone of Cloud, and hence vulnerabilities in network directly affect the security of Cloud [14], thus, the presence of network IDS (NIDS) in a Cloud network plays a vital part in attack mitigation, and it has become an integral part of a secure Cloud. NIDS represents a protection layer that detects real-time aggressive behavior, malicious activity or suspicious pattern by monitored the network traffic and takes corrective measure to avoid or minimize the occurrence of attacks. As result for that, data integrity will be preserved from attacks, security and safety of the network will be maintained, the administrative capacity of the system administrator's security will be reinforced and operational performance of the system will be optimized [15].

In recent years data mining techniques are used for intrusion detection in wide range because the automation of intrusion detection. The pattern of intrusion and normal behavior can be computed using data mining. Intrusion Detection mechanisms (IDS) based on data mining technique are not only automated but are also provided for a significantly elevated accuracy and efficiency. It can help in revealing of new intrusions and policy violations, promoting decision support for intrusion management, and discovering behavior patterns of attackers that unknown previously. One of the data mining techniques that has successful used in solving complex practical problems is neural network. Artificial neural networks have the ability to solve several problems confronted by the other present techniques used in intrusion detection [15]. There are three advantages of intrusion detection based on neural network ([16, 17] :

- Neural network provides elasticity in intrusion detection process, where the neural network has the ability to analyze and ensure that data right or partially right. Likewise, neural network is capable of performing analysis on data in nonlinear fashion.
- Neural network has the ability to process data from a number of sources in a non-linear fashion .this is very important especially when coordinated attack by multiple attackers is conducted against the network.
- Neural network is characterized by high speed in processing data.

According to survey in [18], Back Propagation Neural Network (BPNN) has good detection rate as compared to other neural network techniques.

In this work, BPNN classifier was used to classify attacks and an Improved Genetic Algorithm was employed to optimize and increase the accuracy of this classifier. Genetic Algorithm (GA) is improved through optimization strategies, namely Parallel Processing and Fitness Value Hashing, which reduce execution time, convergence time and save processing power.

Our major purpose is to build an effective and an efficient network intrusion detection system called “ANIDS BPNN-IGA”, based on anomaly approach using a very popular soft computing tool widely used for intrusion detection namely Back propagation Neural Network (BPNN), optimized through an Improved Genetic Algorithm (IGA). The main goal of our developed system is to reduce impact of network attacks (known attacks, and unseen attacks), while ensuring higher detection rate, lower false positive rate, higher accuracy and higher precision with an affordable computational cost. Further, we have chosen to position the proposed IDS on Front-End and Back-End of the Cloud, to detect and stop attacks in real time impairing the security of the Cloud Datacenter.

The rest of this paper is organized as follows: Section 2 discusses different aspects related to Cloud based intrusion detection systems, such intrusions in Cloud, definition of IDS, intrusion detection methods, IDS taxonomy based on their locations within Cloud and the challenges and essential features of Cloud IDS. Section 3 introduces previous research works. Section 4 explains the background of this paper such as Artificial Neural network, BPNN, Genetic Algorithm, Optimization strategies for Genetic Algorithm, KDD Cup' 99 Dataset, Feature Selection and Data Preprocessing. Section 5 presents positions of the proposed system in a Cloud Network, describes our proposed approach and gives an overview of our implementation. Experimental results and analysis are given in section 6. Finally we conclude in section 7.

2. Cloud based Intrusion Detection Systems (Cloud IDS)

This section discusses different aspects related to Cloud based intrusion detection systems, such intrusions in Cloud , definition of IDS, intrusion detection methods, IDSs taxonomy based on their locations within the Cloud, and the challenges and essential features of Cloud IDS.

2.1 Intrusions in cloud

An intrusion is defined, according to NIST (National Institute of Standards and Technology) as an attempt to endanger security policies (privacy, integrity, and availability) or skip computer and network security mechanisms [13]. The action of intrusions gradually weakens the confidentiality of resources and information present or hampers the integrity and availability of behavior in a host or in a network cloud environment. Therefore, intrusions are any sets of activities that violate the security policy of the computer systems and threat the veracity, confidentiality, or accessibility of a network resource. It can also provide unofficial and illegal access to prominent and valuable information and unauthorized file modification which are reasons behind harmful activities [19]. The attacks that may affect cloud computing system are: Insider attack, Denial of service (DOS) attack, User to root attack, Port scanning, Attacks on virtualization, and Backdoor channel attacks.

- **Insider attack:** The business man, employers and partners who work in the present time or in the past time that have authorized access to information of users indicated as insiders [10]. Those insiders may attempt get unauthorized privileged access to cloud resources or misuse the privileges that are either assigned or not assigned to them officially. Consequently, they may commit frauds, modify information intentionally or reveal secrets to opponents. This attack is very hard to detect.

- **Denial of service (DOS) attack:** In this attack also called flooding attack, the attacker sends massive amount of packets from innocent hosts (zombies) in network to flood the victim virtual machine. Also, a hacker can cause denial of system services by consuming the bandwidth of the network by means of the Worms for example, which replicate themselves and spread within minutes to a large number of computers, leading to network congestion [20]. The type of packets can be TCP, UDP, ICMP or a Blend of them. The aim of this attack is to deny access for legitimate users and hack the cloud resources. By attacking a single server providing a certain service, attacker can cause a loss of availability of the intended service. Such an attack is called direct DoS attack. If the server's hardware resources are completely exhausted by processing the flood requests, the other service instances on the same hardware machine are no longer able to perform their intended tasks. Such type of attack is called indirect DoS attack [2].
- **User to root attacks:** User to Root exploits [21] are a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system. There are several different types of User to Root attacks. The most popular type of this attack is buffer overflow. The method of execute buffer overflow attack is when the application program copies amount of data into buffer its size less than data. There are no universal standard security mechanisms that can be used to prevent security risks like weak password recovery workflows, phishing attacks, keyloggers, etc. In case of Cloud, attacker acquires access to valid user's instances which enables him/her for gaining root level access to VMs or host.
- **Port scanning:** Attackers can use port scanning method to obtain list of open, closed, and filtered ports. Through this technique, attackers can determinate the open ports and attack the services running on these ports. Different techniques of port scanning are SYN scanning, ACK scanning, TCP scanning, Windows scanning, FIN scanning, UDP scanning etc. They reveal the entire network related information such MAC address IP address, router and gateway filtering and firewall rules in cloud system, port scanning attack may cause loss of confidentiality and integrity on cloud.
- **Attacks on virtualization:** An attacker may successfully control the virtual machines by compromising the lower layer hypervisor. For e.g. SubVir, BLUEPILL, and DKSM are well-known attacks on virtual layer. Through these attacks, hackers can be able to compromise installed-hypervisor to gain control over the host. Attackers easily target the virtual machines to access them by exploiting the zero-day vulnerabilities in virtual machines, this may damage the several websites based on virtual servers [22].
- **Backdoor channel attacks:** It is a passive attack, which allows hackers to gain remote access to the infected node in order to compromise user confidentiality. Using backdoor channels, hackers can be able to control victim's resources and can make it a zombie for attempting a Does attack. It can also be used to disclose the confidential data of the victim [23] and revealing victim's secrecy. As result, compromised system confronts difficulty in performing its regular tasks. In Cloud environment, attacker can get access and control Cloud user's resources through backdoor channel and make VM as Zombie to launch DoS/DDoS attack. This attack affects confidentiality of the system [2, 24].

2.2 Intrusion detection systems (IDS)

An intrusion can also be defined as ''any set of actions that attempts to compromise the integrity, confidentiality, or availability of a resource'' [25]. To counter intrusions, an intrusion detection system (IDS) comes into action. An IDS is defined, according to the NIST (Institute of Standards and Technology) [13] as a supervision process over computer and network systems while analyzing the intrusion signs. An IDS could be software, hardware or a combination of both that monitors network or system activities for malicious activities or policy violations and notifies network manager by mailing or logging the intrusion event [26].

There are several techniques and frameworks that are implemented in IDS. In general, IDSs are comprised of [27]:

- **A set of sensors:** They gather both normal and malicious data from the monitored system. Sensors may be part of the system or external appliance depends on the type of IDS.
- **An analyzer engine:** It collects all data from sensors and analyses them. The engine usually placed in central point, and it has capability to reconfigure the monitored system properly if the results of the analysis point out an intrusion taken place.
- **A report system:** It notifies the security officer when suspicious events occurred.
- **A response engine:** It is able to take actions automatically or manually by the command of the security officer.

2.3 Intrusion detection methods classification

The detection of intrusions can be performed basing on analyzing the events which occur in the monitored network. Two primary approaches are used by IDS: misuse or signature detection and anomaly detection. To improve the performance of IDS, it is better to use a combination of these techniques, which called Hybrid detection. Figure 1 shows three detection techniques used by IDSs.

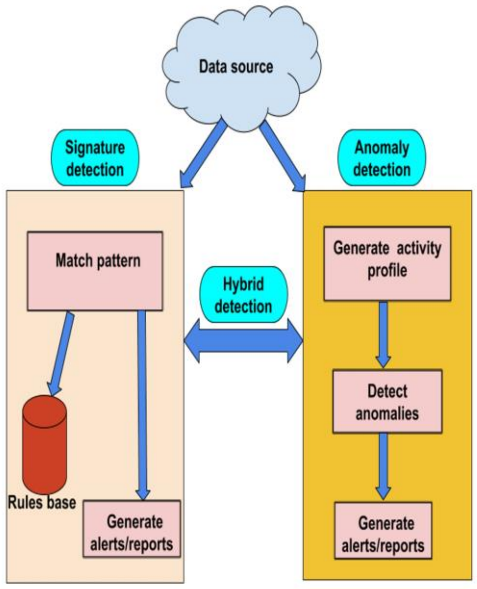


Figure 1. Detection techniques used by IDS

2.3.1 Signature Based Detection

This detection technique exists in the majority of commercial NIDSs [28], aims to detect known attacks by using predefined attack patterns and signatures so it looks for a specific event that has already been recognized and registered. This approach is also recognized as misuse detection or knowledge-based detection. In the signature-based detection, a set of rules (signatures) are produced and maintained in signature/misuse database, and are used to make decision on a received data and attacks detection [2,29] Signature is a pattern (string) corresponding to a known attack or threat. These signatures are composed by several elements that allow identifying the traffic. For instance, in Snort, the parts of a signature are the header (e.g. source address, destination address, ports) and its options (e.g. payload, metadata). To decide whether or not the network traffic corresponds to a known signature, the IDS uses pattern recognition techniques. Some IDS that use this approach are Snort, Network Flight Recorder, Network Security Monitor and Network Intrusion Detection, etc. The advantage of using signature-based detection method is its high accuracy for intrusion detections. In fact, signature-based intrusion detection systems do not generate any inaccurate alarms [6]. Thus, it helps network managers with average security expertise to identify intrusions accurately. Another advantage of this method is the ease of maintenance and updating the predefined/preconfigured rules, and it is a flexible approach since new signatures can be added to database without modifying existing ones. However, it may not always detect an attack with even slight variations [30], and it is unable to detect unknown attacks [31].

2.3.2 Anomaly Based Detection

The anomaly detection builds a model from normal behaviors and any deviation from the normal model is deemed to be an outlier/attack [32]. In other words, this approach consists of comparing current user activities against preloaded profiles of users or networks to detect abnormal or unusual behavior that may be intrusions. The profiles may be dynamic or static and correspond to the expected or legitimate behavior of users. To build a profile, regular activities of users, network connections, or hosts are supervised for a specific period of time called training period [13]. Profiles are developed using different features such failed login attempts, number of times a file is accessed by a particular user over a particular time duration, CPU usage etc. Anomaly detection approach is based on techniques such as: Threshold detection, rule-based measures, statistical measures, machine learning and data mining methods. The first technique expresses some attributes of user and system behavior in terms of counts. Then it compares the latter with a tolerance level. The second approach tries to define a set of rules that can be used to decide whether a given behavior is normal or not. Statistical measures analyze the distribution of the network traffic attributes and can be parametric or non-parametric, the first one is assumed to fit a particular pattern while the second is learned from a set of historical values. The last technique based on machine learning and data mining learns from a set of training data and constructs a model able to classify new network traffic as legitimate or malicious [28]. Anomaly detection methods based on heuristics or rules are able to detect known and unknown attacks. This propriety is very prominent since new kinds of vulnerabilities and intrusions are constantly appearing. However, new legitimate behavior can be falsely identified as attack, resulting in a false positive.

2.3.3 Hybrid Detection

The efficacy of IDS can be significantly improved by combining signature based and anomaly based techniques which is called Hybrid detection technique. The idea behind the implementation of hybrid detection is to exploit benefits of both misuse and anomaly detection techniques, thus, the IDS will be able in order to detect both known and unknown attacks [22]. Recently new hybrid intrusion detection systems are developed and showed great success [33, 34].

2.4 Intrusion detection systems taxonomy based on their locations within cloud

As shown in figure 2, the intrusion detection systems are classified based on their locations in the Cloud computing environment into four categories: host-based IDS, network-based IDS, distributed (decentralized) IDS, and hypervisor-based IDS.

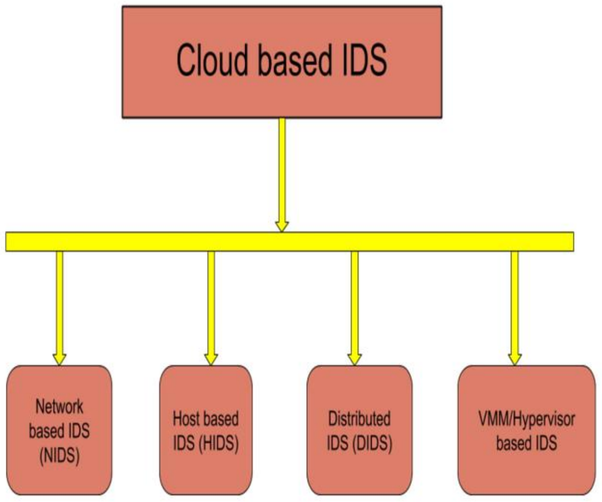


Figure 2. Cloud based IDS

2.4.1 Host based IDS (HIDS)

HIDS monitors and analyzes the information collected from a specific host device machine to detect unauthorized and intrusive events. In host-based IDS, an intelligent agent is installed over the supervised host. This intelligent agent considers different aspects of host security (such as operating system events files and application events files) and analyzes them [35]. HIDS observes modification in host kernel, host file system and behavior of the program. Upon detection of change in behavior or change of system or program, it reports to network manager that system is under attack [36]. The effectiveness of HIDS can be improved by specifying the features that provide it more information for detection. However, it requires more storage for information to be analyzed. In the case of cloud computing network, it is possible to deploy HIDS on hypervisor, VM or host to analyze the system logs, user login information or access control policies and detect intrusion events. Cloud user is responsible for monitoring of HIDS deployed at a VM while cloud provider is responsible for the deployment of HIDS on hypervisor [2] HIDS is capable of analyzing encrypted traffic however; it is susceptible to DoS attack and can even be disabled. HIDS are commonly used to protect the integrity of software.

2.4.2 Network based IDS (NIDS)

NIDS is a tool which monitors and analyses data traffic on the network to detect possible threats to the network or identify intrusive activities (such DoS attacks, port scanning, user to root attacks etc.) and raises an alarm whenever a suspicious activity is detected [37,38] **.** It usually performs intrusion detection by inspecting the IP and transport layer headers of each packet. NIDS utilizes the anomaly and/or signature based detection approach to identify intrusions. For signature detection approach, it looks for the correlation of captured packet with signatures of known attacks, while for anomaly detection method; it evaluates monitoring data against normal baseline and will issue an alarm if there is an occurrence of the abnormal behavior [39]. The cloud computing components could be classified to front-end and back-end, according to their locations. Placing NIDS in the front-end of the cloud helps the detection of the attacks and intrusions from the external network. However, this is not sufficient, because the system may not be able to detect internal intrusions [6]. To overcome this issue, positioning NIDS also on processing severs helps to detect intrusions at internal network of the Cloud by monitoring virtual network and the flow of traffic from or to the processing servers on the physical network [40].

2.4.3 Distributed IDS (DIDS)

A Distributed IDS (DIDS) contains numerous IDSs (such as NIDS, HIDS) that are deployed over a large network to monitor and analyze the traffic for intrusive detection behavior. The participant IDSs can communicate with each other or with a centralized server. Each of these individual IDSs has its own two function components: detection component and correlation manager. Detection component examines system’s behavior and transmits the collected information in a standard format to the correlation manager. Correlation manager combines data from multiple IDS and generates high level alerts that keep up a correspondence to an attack. Analysis phase makes use of signature based and anomaly based detection techniques so DIDS can detect known as well as unknown attacks. In case of cloud, DIDS can be located at any of two positions: at processing server or at Host machine [11].

2.4.4 Hypervisor-based Intrusion Detection System

Hypervisor provides a platform to run VMs. Hypervisor based IDS is deployed at the hypervisor layer. It allows monitoring and analyzing of available information for detection of anomalous activities and events. The information is based on communication at different levels like communication between VMs and communication within the hypervisor based virtual network [31].

2.5 Challenges and essential characteristics of cloud IDS

In this section, we present the main challenges to Cloud ID. Thus, we give essential characteristics of Cloud IDS to be an effective IDS that can detect intrusions in traditional as well as virtual network in Cloud Environment, while reducing false positives and false negatives, with affordable computational cost and higher detection accuracy.

2.5.1 Challenges to Cloud IDS

Cloud IDS has some main challenges that are as follows:

- **Attacks on virtual environment:** In a virtualized environment, VMs communicate over hardware backplane rather than a network. As a result, the standard network security controls are blind to this traffic, and cannot execute security control for supervising and in-line blocking. In fact, a malicious user having a VM instance can perform several attacks like (Rubens, 2010; Kenneth, 2010) Hyper jacking, VM escape, VM hopping, VM migration to gain control of other’s VM or host machine. Cloud NIDS should be able of monitoring and detecting intrusions from network traffic between VM and the host.
- **High network traffic:** Not long ago, Cloud is a rapidly growing computing model that provides various advantages in economic and business aspects. Thus, Cloud users are augmenting at very high rate. This generates heavy network traffic from a great number of Cloud users. IDS should handle such traffic quickly. Otherwise, it will be resulting into high probability of packet dropping.
- **NIDS deployment:** In Cloud, the major challenge is to monitor both external and internal traffic for securing and protecting front end and back end of Cloud. This is due to the distributed and visualized nature of Cloud. Therefore, IDS should be deployed in the manner that they can detect internal attacks, external attacks and distributed attacks like DDoS attacks in the overall Cloud network.

Moreover, traditional IDS challenges should be considered before integrating IDS in Cloud environment such false positives, false negatives, detection accuracy and detection rate.

2.5.2 Essential Characteristics of IDS for the Cloud

IDS should have the following characteristics for integrating it in the cloud:

- **Detection of attacks on each layer:** IDS should be able to detect intrusions at each component of Cloud architecture, either at the front end or at the back end. It should be capable of detecting known attacks as well as unknown attacks.

- **Low computational cost and faster detection rate:** In Cloud, great number of users is involved. So, high number of requests may turn into high traffic rate in Cloud. Thus, IDS should have faster detection at lower cost.
- **Low false positives and low false negatives:** The term false positive describes a situation, in which an IDS triggers a false alarm, but it is a wrong alarm, in fact, there is no attack. Whereas, false negative can be defined as an inability of IDS to detect the true intrusion; in other words, malicious activity is not detected or alerted. The researchers need to keep very low false negatives and false positives in the Cloud, to let pass legitimate network packets and to protect network against malicious traffic. Fortunately, there are some actions that can be taken to reduce the chance of false negative conditions without increasing the number of false positives [41, 42].

3. Literature Review

Modi et al. [43] have integrated a signature Apriori based NIDS to Cloud. Signature Apriori takes network packets and known attack signatures as input and generates new derived rules that are updated in the Snort. Therefore, Snort is able to detect known attacks and derivative of known attacks in the Cloud. This approach improves the efficiency of Snort. However, it cannot detect unknown attacks.

Modi and Patel [44] have proposed a hybrid-network intrusion detection system (H-NIDS) deployed on each host machine, to detect internal and external network attacks in Cloud Computing environment. The architecture of proposed H-NIDS consists of mainly seven successive modules; Packet capture, Signature based detection, Anomaly detection, Score function, Alert system and Central log. Signature based detection module uses Snort and signature Apriori algorithm, which generates derived attack rules, thereby, Snorts can detect known attacks and derivative attacks. Anomaly based detection module uses a combination of multiple classifiers; Bayesian, Associative rule and Decision tree. They predict the class label of given network packets and send the result to score function. The score function uses weighted averaging method to determine whether the predicted intrusion is really intrusion or not. Moreover, it determines whether the detected intrusion is a type of distributed attack or not, by checking the central log of malicious packets and applying a majority vote method. The Alert system raises alerts about intrusion that is determined either by Sort or score function, and stores alerted intrusion in the central log base. Further to the experiments realized by the authors, proposed H-NIDS has capability of detecting known and known attacks efficiently with high accuracy and low false alerts. Moreover, that system has lower computational and communication overhead than agent based approaches.

Moorthy and Rajeswari [45] have proposed security architecture for cloud, in which a virtual host based intrusion detection system was placed between router and Cloud host. The developed IDS consists of three components namely: Event Auditor, IDS service (combination of analyze system and Alert system) and CIDD (Cloud Intrusion Detection Data Sets). The analyzer system examines the content of packet against the cloud intrusion datasets signatures stored in CIDD by means of pattern matching. The experiments conducted by the authors show that the proposed IDS was able to detect 80% of random sets of cloud attacks and no false positive alarm is raised while filtering background traffic received form DARPA dataset. However, results show that latency in IDS is increasing according to background traffic, and a breaking point was identified at 2 mbps, in which, the IDS generated an error and stopped. Therefore, an unstable interval was determined between 1.5 to 2 mbps.

Mehmood et al. [46] have proposed a Distributed Intrusion Detection System using Mobile Agents in Cloud Computing (DIDMACC) to detect distributed attacks in Cloud. They have used mobile agents to carry intrusion alerts collected from different VMs where Suricata NIDS is deployed to the management server. In this server, the correlation module (Open Source Security Information Management (OSSIM) correlation engine) correlates intrusion alerts to generate high level alerts that correspond to a distributed attack. Then, the management server sends the signature of detected attack to all virtual machines monitored, to update the signature database of local Suricata IDS to avoid such intrusions in future. The results show that the use of mobile agents to carry intrusion-related data and code reduces network load, and correlation of intrusive events collected by those mobile agents by means of a correlation engine helps in detection of distributed intrusions. However, the proposed system can't detect zero-day attacks or unknown attacks.

Sangeetha et al. [47] have proposed a Signature based Semantic Intrusion Detection System on Cloud, which concentrates on the application level to detect application specific attacks. Those attacks aim to compromise the system by exploiting vulnerabilities of the protocols of the application layer such as HTTP, FTP etc...The packets transferred between cloud users and servers are captured by Cloud IDS Engine and analyzed for any maliciousness. The operation of the components of that IDS is as follow; the packets of various protocols captured by packet sniffer are forwarded to the protocol analyzer, which recognizes the protocol type and dispatches them to its corresponding parser. The parser translates a sequence of packets into protocol messages and forwards them to the parsing grammar for analysis and checking with the semantic rules. Semantic Classification tree is constructed by analyzing the specification of the protocol. The specification gives the rules and the individual patterns which will be matched in the corresponding fields of the protocol. The tree is formed in the top-down format. As each node on the path from the root to a leaf node checks with the input, if any signature does not matches with the rule base then it raises alerts to the cloud IDS Interpreter which in turn alerts the Virtual Cloud Provider. The traffic is continuously monitored and analyzed for any malicious behavior and is reported to the administrator. Even though it is a novel approach, however, the implementation details and the results are not given to prove the concept.

Singh et al. [48] has proposed a novel Collaborative IDS (CIDS) framework for cloud, to defend network accessible Cloud resources and services from various threats and attacks. The proposed NIDS is integrated in each cloud cluster, and a correlation Unit (CU) provides collaboration between all cluster NIDSs, is placed in any one cluster. Bully election algorithm is used to elect one best cluster for placement of CU on the basis of workload. The hybrid NIDS use Snort to detect the known stealthy attacks using signature matching, and to detect unknown attacks, anomaly detection system (ADS) is built using Decision Tree Classifier and Support Vector Machine (SVM). In the proposed model, cascading decision tree and SVM has improved the detection accuracy and system performance as they remove the limitation of each other. Use of DT makes the learning process speedy and divides the dataset into small sub datasets. Use of SVM on each sub dataset reduce the learning time of SVM and overcome the over-fitting and reduce the size of decision tree to make the detection faster. For frequent attacks detected by ADS, signature generation process generates a Snort based signature. Once the signature is generated, local knowledge base is updated, and this signature is sent to the central correlation unit. It receives the signature sent by all the NIDSs in the Cloud network and calculates the value of a criterion to make a decision on the bases of how much part of total NIDSs send the similar signature. If calculated value for an attack signature is higher than a threshold, thereafter, correlation unit multicasts this signature to all the IDSs. They receive this signature and update their knowledge base. By this way, collaboration between NIDSs prevents the coordinated attacks against cloud infrastructure and knowledge base remains up-to-date. The performed experiments by the authors show that the proposed ADS outperform both SVM and decision tree in terms of accuracy and computation time when they are used separately.

Saljoughi et al. [8] have presented a network intrusion detection system (NIDS) for Cloud environment using Multilayer Perceptron Neural Network (MLP) and Particle Swarm Optimization Algorithm (PSO) to detect intrusions and attacks. The PSO algorithm was utilized to find the best weights and biases of the neural network (MLP), which is then trained by trained data and the obtained optimal weights. In order to have the most efficiency and security, the proposed NIDS is placed in the network, and it is connected directly to the router of the Cloud, and the others similar NIDS are installed on the processing servers. All NIDS send attack incidents to a central server with a large storage space; and if necessary, this data will be used by the proposed system. The results obtained from optimization of the neural network using the Particle Swarm algorithm showed a substantial improvement in the function of the NIDS based on MLP, in terms of the precision of detecting attacks faced by the networks and reduction of time complexities.

Hatef et al. [6] have proposed a hybrid network intrusion detection system called HIDCC, which combines signature-based detection technique and anomaly-based detection technique, in order to identify efficiently the internal and external attacks in the cloud environment. Snort was used as a signature-based intrusion detection module to detect known attacks by using the known attacks rules database and derived attacks database. For trapping unknown attacks that were not detected by Snort, the anomaly detection module built by combination of learning vector quantization algorithm and C4.5 algorithm comes into play to detect those attacks. If the input packet received by that module is detected as an unknown attack, the packet is removed and a warning is generated. Therefore, the pattern of this attack is added to the known attacks database so that in the following step, when a similar attack occurs, snort would be able to detect it immediately and prevent it from penetrating the system. Further, the Apriori algorithm was used to generate a pattern for the derived attacks to update derived attack database. The implementation results of the proposed method show that the intrusion coverage, intrusion detection accuracy, availability, and reliability in cloud computing systems are considerably boosted and false alarms are significantly reduced.

Ghosh et al. [30] have designed a network intrusion detection system to detect attacks and malicious activities in Cloud environment. The proposed IDS includes two stages. The first stage consists of creation of a feature subset by using a novel algorithm called BCS-GA which combines the advantages of Binary Cuckoo search algorithm (BCS) and Genetic Algorithm (GA). The proposed BCS-GA algorithm was applied on NSL-KDD training dataset to remove several irrelevant features, in order to reduce the training time and memory storage space required for such high dimensional dataset. Thus, initial NSL-KDD dataset contains 41 features, but after applying BCS-GA algorithm, it successfully reduced to 16 features. In the second stage, Neural Network classifier was trained by the reduced training dataset using 16 features. Thereafter, classification accuracy of that classifier was tested by means of a separate reduced testing dataset. Experimental results indicate that the proposed IDS produces 78.229% of accuracy.

Mehibs and Hachim [15] have proposed Back Propagation Artificial Neural Network to build network intrusion detection system with the goal to detect intruders and suspicious activities in and around the cloud environment. The proposed module consists of two stages, the learning stage and the test stage. In the first stage, this model is trained with back propagation algorithm using KDD 99 dataset to classify normal behavior and the other four types of attack (DOS, Probe, U2R, and R2L). In the second stage, the trained module is evaluated with three datasets to predict the class label of test samples. The topology adopted for neural network of proposed IDS consists of three layers (input layer, hidden layer, output layer); the number of neurons in input layer is equal to 41 which is the number of feature in KDD99 dataset. The number of neurons in hidden layer is equal to 20, which is determined after trial and error. In the output layer, the number of neuron is equal to 5 which correspond to the normal behavior and the four types of attack. The experimental result demonstrates effectiveness of the proposed NIDS characterized by high detection rate and low false alarm.

4. Related Background

This section provides the necessary background to understand the problem in hand. Subsections 4.1, 4.2, 4.3 and 4.4 shed the light on Artificial Neural Network, Back Propagation Neural Network, Genetic Algorithm and Optimization Strategies for Genetic Algorithm. Section 4.5 describes KDD cup 99 Dataset. Next, section 4.6 briefly presents Feature Selection technique, and Section 4.7 defines the two operations that make up Data Preprocessing, namely Categorical encoding and normalization.

4.1 Artificial neural network

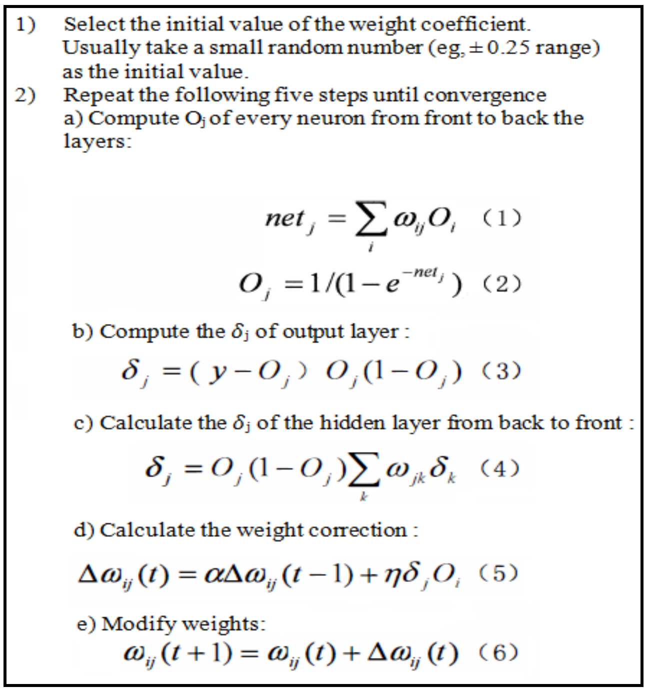
Machine learning (ML) is the type of the data mining technique. The machine learning technique is basically used to find the data with relationships, and then, it analyses the process for extraction. The ML technique is categorized into 3 classes, i.e. SL (Supervised Learning), RL (Reinforcement Learning) and UL (Unsupervised Learning). The SL is a learning technique which is learned with the guidance. In other words, a supervisor or a teacher helps in learning process. The UL is a learning technique which is performed without any guidance. In other words, there is no help provided in the learning process. The Artificial Neural Network is the type of ML technique [1]. An artificial neural network (ANN) is a computational model inspired from biological nervous systems; it mimics the neurons of human brain. It consists of set of simple processing components called artificial neurons that are interconnected to other neurons by synapses or links. Each link is associated with weight. The neurons are arranged in layers and connected through weighted connections (links); each neuron represents a computational unit that receives input to process it to get the output. The connections set the direction of information outflow, which can be unidirectional or bidirectional [15]. The aim of ANN is to find the desired outputs from the inputs. It is capable to learn things by itself [49]. The process of ANNs learning is based on examples. In the learning phase, the network of ANN adapts to find the desired outputs from the inputs by modifying the weights of connections between the nodes. In the neural network, the method used to adjust the weights and the way (topology) that nodes are connected determine the type of neural network [17]. Further, The ANNs have the ability to overcome incomplete, noisy and limited data. Thus, ANN has been acknowledged as an intelligent universal mechanism of dealing with function approximation, pattern recognition, process estimation and prediction, optimization design and other real applications [50]. The aim of using Neural Network for intrusion detection is to classify data as normal or attack. The ability of high tolerance makes the Neural Network flexible and powerful in IDS [51]. Briefly, the advantages of using Neural Networks are [52]:

- Neural Networks are data driven self-adaptive methods in that they can adjust themselves to the data without any explicit specification of functional or distributional form for the underlying model.
- Neural Network has the capability of analysing the incomplete and distorted data from the network.
- Neural Networks are universal functional approximators. The Neural Networks can approximate any function with arbitrary accuracy. Since any classification procedure attempts a functional relationship between the group membership and the attributes of the object, accurate identification of this underlying function is important.
- Neural Networks are non linear models, which makes them flexible in modeling real world complex relationships.
- Neural Networks are able to estimate the posterior probabilities, which provide the basis for establishing classification rule and performing statistical analysis.

4.2 Back propagation neural network (BPNN)

Back Propagation Neural Network (BPNN) is a special type of neural network [53]. It is also named error back propagation neural network, and it is a multilayer feed forward neural network which uses Multi-layer Perception MLP) as network architecture and Back Propagation Learning Algorithm as training or learning algorithm. Back Propagation network learns by example. You provide the algorithm examples of what you want the network to do and it changes the network's weights so that, when training is achieved, it will give you the required output for a particular input [32]. BPNN with a strong self-learning ability, be able to do adaptive calculations, is a large scale nonlinear adaptive systems. For every kinds of neural networks, BPNN is relatively mature [49]. BP network is widely used in the current pattern recognition, mapping tasks, process control, image processing and analysis, mechanical fault diagnosis, expert systems and other relative fields [54]. A BPNN has multiple layers. Each layer consists of one or multiples neurons (nodes) with some 'activation function' like Hyperbolic tangent or Sigmoid. The left-most layer is known as 'Input layer' and the right-most layer is known as 'Output layer'. Between these two layers there may be one or more than one 'Hidden layers'. In BP network, each neuron at any layer is connected to all neurons in the previous layers. Such layers are called fully connected networks. The input layer is a transmitter and a means of providing data and the output of that layer is transferred and becomes the input of the hidden layer(s). The middle, hidden, layers are transmitter and processor where data is processed then the output of the hidden layers becomes the input of the output layer. Finally, from the output layer it is gotten the output data, which contains the values predicted by the network [55, 15].

BPNN as supervised machine learning technique is composed of the two successive stages; the learning stage and the testing stage. During the first stage, the network is learned by modification of the weights. During the second stage; the weights from learning stage are used to forecast the class label of the new input patterns [15]. As shown in figure 4, the learning process of BPNN can be divided into two parts: the forward propagation process and the error back propagation process. During the former one, the output of each neuron is based on the output values and corresponding weights of all the neurons in the previous layer, the bias factor, and the activation function. During the back propagation process, based on the forward propagation, the result of the calculation is compared with the expected result, and then inverses the error, and correct the weights and bias [56]. Every set of forward and backward operations are termed as single 'Epoch'. For every epoch, a fresh set of patterns is given to the network as inputs. The network is trained in this way with a training set for certain number of epochs. After the training phase, the network is capable of identifying the unknown pattern according to its training [57]. Several research papers have used neural network approach [17]. According to survey in [18], BPNN has good detection rate as compared to other neural network techniques.



1) Select the initial value of the weight coefficient. Usually take a small random number (eg. ±0.25 range) as the initial value.
2) Repeat the following five steps until convergence
   a) Compute O_j of every neuron from front to back the layers:

$$net_j = \sum_i \omega_{ij} O_i \quad (1)$$

$$O_j = 1/(1 - e^{-net_j}) \quad (2)$$

   b) Compute the δ_j of output layer :

$$\delta_j = (y - O_j)\ O_j(1 - O_j) \quad (3)$$

   c) Calculate the δ_j of the hidden layer from back to front :

$$\delta_j = O_j(1 - O_j)\sum_k \omega_{jk}\delta_k \quad (4)$$

   d) Calculate the weight correction :

$$\Delta\omega_{ij}(t) = \alpha\Delta\omega_{ij}(t-1) + \eta\delta_j O_i \quad (5)$$

   e) Modify weights:

$$\omega_{ij}(t+1) = \omega_{ij}(t) + \Delta\omega_{ij}(t) \quad (6)$$

Figure 3. Operation of BPNN learning process

4.2.1 Role of Learning Rate and Momentum Term in Learning Phase of BPNN

As it is shown in figure 3 above, to adjust the weights of connections between neurons in neural network, Back Propagation Learning Algorithm uses equations (5) and (6). In those equations, this algorithm uses two prominent and crucial parameters namely Learning rate and Momentum term.

- **Learning rate (η):** The learning rate is a relatively small constant that indicates the relative modification in weights. If the learning rate is too low, the network will learn very slowly, and if the learning rate is too high, the network may oscillate around minimum point, overshooting the lowest point with each weight adjustment, but never actually reaching it. Habitually, the learning rate is very small, located in the interval [0; 1].
- **Momentum term (α):** The introduction of the momentum term is used to accelerate the learning process by "encouraging" the weight changes to continue in the same direction with larger steps. Furthermore, the momentum term prevents the learning process from settling in a local minimum by "over stepping" the small "hill". Typically, the momentum term has a value between 0 and 1 [58].

4.3 Genetic algorithm

John Holland [59] in 1970s introduced familiar problem solving algorithms called Genetic Algorithms (GAs) which is based on the principles of biological development, natural selection and genetic recombination. GAs are computational intelligence techniques and search procedures often used for optimization problems [60]. The solution to the problem is encoded in chromosome (each chromosome represents an individual) like data structure. Each parameter in a chromosome is called as gene. Genes are selected according to our problem definition. These are encoded on bits, character or numbers. The set of generated chromosome is called a population [61]. An evaluation function is used to calculate the goodness of each chromosome according to the desired solution; this function is known as "Fitness Function". GA evolves the group of chromosomes to a population of quality individuals through: selection (usually random selection), cross-over (recombination to produce new chromosomes), and mutation operators.

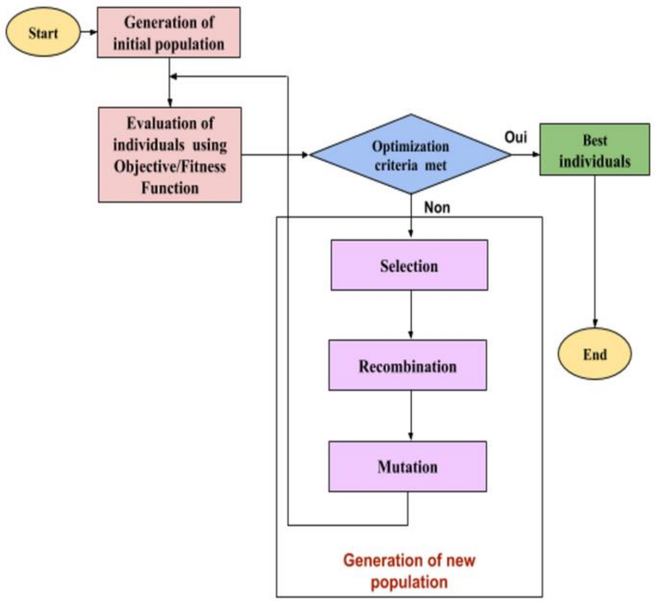

Start
Generation of initial population
Evaluation of individuals using Objective/Fitness Function
Optimization criteria met
Oui
Best individuals
Non
Selection
Recombination
Mutation
End
Generation of new population

Figure 4. Flow of Genetic Algorithm

Figure 4 illustrates the process of a genetic algorithm. GA process begins with series of initial solutions is initially generated (random population) and through a combination of algorithms similar to an evolutionary process (often a combination of elitism, crossover, and mutation) the process works towards evolving solutions having better "goodness" as evaluated by the fitness function. In every generation, the fitness of these chromosomes is checked. To determine the fitness of the chromosomes fitness function is used and then fittest chromosomes are selected. The chromosomes which have poor fitness value are discarded. The selected fit chromosomes undergo crossover, mutation to form a new population. This new population is used for the next generation. Normally, the algorithm terminates when either a set number of generations or a satisfactory fitness level has been achieved. Genetic algorithm is composed of three operators. They are reproduction or selection, crossover or recombination and mutation [62].

GA is very promising in the computer security field, especially in IDSs. It has been applied for intrusion detection since the 1990's, and is still being used up till the current time. In general, GA can be involved in many tasks in IDSs, such as optimization, automatic model design, and in classification. GA is usually used to generate rules (build a classifier) used to detect anomalies, and they usually take the form if {condition} then {action}, where the condition part test the fields of incoming network connections to detect the anomalous ones [63]. When a GA is used for problem-solving, three factors will have impact on the effectiveness of the algorithm, they are: 1) the selection of fitness function; 2) the representation of individuals; and 3) the values of the GA parameters [64].

4.4 Optimization strategies for genetic algorithm

With the fitness function, typically being the most processing demanding component of genetic algorithm (GA), it makes sense to focus on improvement of the fitness function to see the best return in performance. In this subsection, we will explore two optimization strategies that are used in this work to improve performance of GA by optimizing the fitness function, namely *Parallel Processing* and *Fitness Value Hashing.*

4.4.1 Parallel Processing

One of the easiest approaches to achieve a performance enhancement of GA is by optimizing the fitness function. The fitness function is typically the most computationally expensive component; and it is often going to be the bottleneck of GA. This makes it an ideal candidate for multi-core optimization. By using multiple cores, it’s possible to compute the fitness of numerous individuals simultaneously, which makes a tremendous difference when there are often hundreds of individuals to evaluate per population. Java 8 provides some very useful libraries that make supporting parallel processing in our GA much easier. Using *Java’s IntStream*, we can implement parallel processing in our fitness function without worrying about the fine details of parallel processing (such as the number of cores we need to support); it will instead create an optimal number of threads depending on the number of cores available in our multi-core system. Hence, by using parallel processing, fitness function will be able to run across multiple cores of the computer, consequently, it is possible to considerably reduce the amount of time the GA spends evaluating individuals and, so reduce the overall time of execution of GA, and accelerate convergence process [65].

4.4.2 Fitness Value Hashing

Fitness Value Fitness Value Hashing is another strategy that can reduce the amount of time spent computing fitness values by storing previously calculated fitness values in a hash table [65]. During running of GA, solutions found previously will occasionally be revisited due to the random mutations and recombinations of individuals. This occasional revisiting of solutions becomes more common as GA converges and begins to find solutions in an increasingly smaller area of the search space. Each time a solution is revisited its fitness value needs to be recalculated, wasting processing power on recurrent, duplicate computations. Luckily, this can be easily fixed by storing fitness values in a hash table after they have been computed. When a previously visited solution is revisited, its fitness value can be retrieved from the hash table, avoiding the need to recalculate it.

4.5 KDDcup 99 dataset

KDD is the abbreviation of knowledge discovery in databases. KDD refers to the overall process of recovering knowledge from data [66]. Specifically, The KDD cup 99 dataset was benchmark dataset used in “The Third International Knowledge Discovery and Data Mining Tools Competition” that is selected from DARPA98 network traffic dataset in 1999 by collecting single TCP dump into TCP connections. The KDD cup 99 dataset is considered the most a favorable and developed standard dataset used in research of evaluating algorithms used in intrusion detection [10]. This dataset is available in tcpdump format. The original tcpdump files were preprocessed for utilization of intrusion detection benchmark. In KDD 99 dataset each TCP connection consists of 41 attribute with label which determine the status of connection either being normal or specific attack type.

The 41 feature are divided into numeric features (38) and symbolic features (3) and are classified into the following classes [10, 67]:

- **Basic features:** These features are taken from TCP/IP packet headers and they describe each single TCP connection. The number of these features is equal to 9 (example: duration, protocol_type, service etc).
- **Content features:** These features are used to evaluate the payload of the original TCP packet and looks for suspicious behavior in the payload portion [68]. The number of these features is equal to 13 (example: logged_in, root_shell, is_hot_login etc).
- **Time-based traffic features:** These features are calculated according to window interval by capturing properties of the connections in the past 2 second that have the same service as the current connection. The number of these features is equal to 9 (example: srv_serror_rate, srv_rerror_rate).
- **Host-based traffic features:** Among four types of attacks in KDD 99 dataset, few attacks span longer than 2 seconds intervals. Host-based traffic features are designed to access all attacks which span longer than 2 second intervals that have the same destination host as the current connection. The number of these features is equal to 10 (example: serror_rate, rerror_rate).

KDD dataset mainly contains four types of attacks. They are given below:

- **Denial of Service attack (DoS):** In this type of attack, attackers/intruders attempt to prohibit authorized users from access over a machine or using services of that machine by making computing and memory resources too busy or by consuming the resources of network. An example of DoS attack is smurf.
- **Probe:** The attacker scans a network of computers to gather necessary information about the target system and find fragility or potential weakness. Then use this fragility to attack this system at later time. An example of probing attack is nmap.
- **Remote to Local attack (R2L):** In R2L attack, an attacker/hacker wants to gain the local access as a user of particular machine without any account. To accomplish this, attacker sends packets to the remote victim machine through network. After that, he exploits fragility or some vulnerabilities to get unauthorized local access to that machine. An example of R2L attack is guess_passwd.
- **User-to-Root (U2R) attack:** In this attack, hacker in the first gets access to normal user then exploits fragility in the system to get root level access .The aim of this attack to get illegitimate super-user privileges. An example of U2R attack is buffer_overflow.

The KDD cup 99 datasets consist of training and test set .As shown in table 5, there are 4,940,000 data sample in the training set, these samples are distributed between normal behavior and 24 attacks (DoS, Probe, U2R, R2L). Whereas, there are 311,029 data samples comprise normal network traffic and 38 types of attack, 24 attacks existed in training set in addition to 14 new attacks. As the training set contain large number of data samples, other training set formed include 10% of data samples used in wide range [68].

4.6 Feature selection

Feature selection is a dimensional reduction process which reduces the number of irrelevant, redundant and noisy features from the dataset or eliminates them as much as possible, and selects significant features in order to allow learning algorithms to act faster and more efficient, and improve the detection accuracy [67, 8]. Many researchers [69, 70] have identified that the presence of irrelevant features may have a negative impact on the performance of learning systems. The predictive accuracy of a particular target concept might not be affected by the irrelevant features. Redundant features are those features that, although relevant to a target concept, provide information that is already provided by another features and therefore do not contribute toward prediction [71]. In other words, redundant features are those features which closely correlated with one or more features and provide no more information than the selected relevant features.

For building a fast and accurate Intrusion Detection System [67], effective feature selection is very prominent. In fact, the existence of irrelevant/redundant features in the network traffic data causes wide range of problems in network traffic classification. Such problems vary from slowing the classifier (IDS) performance to restraining a classifier from making error-free decisions [72]. Feature selection has been successfully used with intrusion detection for different objectives: improving the detection rate and the performance of the classification, decreasing storing of memory space, reducing training time [73], decreasing false alarms and decreasing the computation demands of the system by removing irrelevant/redundant features from the input data. As a result, a new set of features that consists of only a subset of the original features will be selected [74]. The intrusion detection core classifier then will be restricted to deal with the selected features of the input data instead of dealing with the entire set of features.

Feature selection algorithms can be categorized into filters and wrappers [75]. Filter methods select subset of features as a preprocessing step, independent of the induction (learning) algorithm. Wrappers use the classifier (learning machine) performance to assess the goodness of feature subsets. Various criteria have been used for evaluating the goodness of a feature including distance measures, dependency measures, consistency measures, information measures, and classification error measures.

In this work, a Kolmogorov-Smirnov correlation based filter was utilized to select a subclass of useful features in intrusion detection. According to this approach, the features highly correlated with the class, but not with one another, were the best features. In this approach, a fast redundancy removal filter based on modified Kolmogorov-Smirnov correlation (KS-CBF) was employed to select features using class information labels while comparing the two features. As shown in table 6, 12 most proper features for intrusion detection were selected [76, 77].

These 12 features are described below [8]:

1. **Service:** Network services at the target.
2. **SRC_bytes:** The number of bytes to move from source to target.
3. **Dst_bytes:** The number of bytes to move from target to source.
4. **Logged_in:** If they are successfully logged in, they are given 1 and are 0 otherwise.
5. **Count:** The number of all connections with the same host as the connection present in the past two sections.
6. **Srv_count:** The number of connections to similar services as the link existed in the past two seconds.
7. **Serror_rate:** The percentage of connections with the “SYN” error.
8. **Srv_rerror_rate:** The percentage of connections with the “REG” error.
9. **Srv_diff_host_rate:** The percentage of connections to similar hosts.
10. **Dst_host_count**: The number of connections to similar hosts.
11. **Dst_host_srv_count:** A series of connections to the similar target port.
12. **Dst_host_diff_srv_rate:** The percentage of connections in various services, among the connections collected in Dst_host_count.

4.7 Data preprocessing

After application of feature selection algorithm on KDD dataset, and consequently selecting of significant features, raw data extracted contained only those selected features needs to be preprocessed before fed into any learning model [12]. Preprocessing [8] includes two operations: data conversion (encoding) and normalization.

- **Categorical encoding:** In the data conversion (encoding) step, when the value of some features is not detectable for the learning algorithm (classifier), i.e., BP neural network, they must be converted to the features with detectable values for the classifier [8]. This operation is called Categorical encoding. Categorical encoding refers to the process of assigning numeric values to nonnumeric features/attributes so as to make the processing task much simpler, as numeric data can be easily handled upon. We consider KDD dataset with 41 features, out of which we have chosen 12 attributes set (01 nonnumeric attribute + 11 numeric attributes) for intrusion detection; we have one nonnumeric attribute namely “Service”. The “Service” feature is nonnumeric and includes http, smtp, finger and etc. Thus, in order to convert these values, numeric equivalences are assigned; for example, http=1, smtp=2, finger=3, etc. Then, these new values will replace the nonnumeric ones.
- **Normalization:** The attributes with high values can dominate the results than the attributes with lower values .This dominance can be reduced by the process of normalization, i.e., scaling the values within certain range [78]. In fact, the main advantage of normalization is to avoid attributes in greater numeric ranges dominate those

in smaller numeric ranges [28]. Normalization is defined as is the process of enclosing the values of attributes to a specific range to minimize the complexity involved in handling data spread over an absolute range and type of values. Various features have values spread over large ranges and types. Hence, they are to be minimized to a specific range being useful to enable proper processing and analysis of the data [79].There are several technique of normalization such as mean-range normalization, frequency normalization, maximize normalization, rational normalization, ordinal normalization, softmax scaling , and statistical normalization [12]. The mean range [0, 1] (Min Max normalization) normalization technique used in this work yields better results in terms of time and classification rate [80]. It is a linear transformation that scales data between [0, 1]. Further, data normalization improves the efficiency and the accuracy of mining algorithms including ANN's, and specially, when the data to be analyzed fall between [0, 1], these algorithms provide better result [15].

The Equation 1 is used by Min Max normalization to find the new value:

$$X' = \frac{x - MinA}{MaxA - MinA} \quad (1)$$

x and x' are value to be normalized and the normalized attribute value respectively. MinA and MaxA are the minimum and maximum possible values for attribute A before normalization.

5. The Proposed System

This section gives positions of the proposed IDS in a cloud network, provides an overview of our previous IDS, describes in detail our proposed IDS and gives the model of that IDS.

5.1 Positions of the proposed system in a cloud Network

The goal of the proposed ANIDS BPNN-IGA is to detect intruders and suspicious activities in and around the cloud computing environment by monitoring network traffic, while maintaining confidentiality, availability, integrity and performance of cloud resources and offered services. It allows detecting and stopping attacks in real time impairing the security of the Cloud Datacenter. As shown in figure 12, we propose to deploy our NIDS on two strategic positions:

- **Front-End of Cloud:** Placing NIDS on front end of Cloud helps to detect network intrusions or attacks coming from external network of Cloud, launched from zombie hosts or by hackers connected to the Internet who attempt to bypass the firewall in order to access the internal cloud, which can be a private one. Therefore, NIDS plays the role of the second line of defense behind the firewall to overcome its limitations [14], and acts as an additional preventive layer of security [81].
- **Back-End of Cloud:** Positioning NIDS sensors on processing servers located at back end of Cloud helps to detect intrusions occurring on its internal network. In a virtual environment, we have many virtual machines on the same physical server, and they can inter-communicate through the virtual switch without leaving the physical server. Thus, network security devices on the LAN can't monitor this network traffic; if the traffic does not need to pass through security appliances primarily a firewall, therefore, a loophole for all kinds of security attacks will be opened. Thus, the starting point of an attacker/hacker is compromising only one VM, and using it as a springboard to take control of the other VMs within the same hypervisor. This is generally done without being monitored or detected, giving the attacker a huge hack domain. Moreover, the virtual environment is exposed to various threats and risks, centered mostly on the hypervisor: Hyper jacking, VM escape, VM migration, VM theft and Inter-VM traffic.

Our NIDS is designed to monitor that virtual traffic, and also the flow of traffic from or to the processing server on the physical network. We haven't chosen to install the NIDS on each virtual machine because it will be an additional burden; it will weigh down the work of the VM. Further, such configuration requires multiple instances of NIDS, which makes complex management of NIDS whereas VMs are dynamically migrated, provisioned or de-provisioned.

5.2 The Overview of our previous IDS

In our previous work [32], we have built an effective Anomaly Network Intrusion Detection System (ANIDS) based on Back Propagation Neural Network (BPNN), which yields higher accuracy, higher detection rate and lower false positive rate in comparison to several traditional and new techniques as demonstrated by experimental results obtained by using KDD Cup'99 dataset.

The two keys of our success were; firstly, we have adopted a novel architecture of neural network, which is given by a new technique of calculating the number of nodes in hidden layer of BPNN (H = 0.75 * Input + Output). Regarding our experiments conducted, this method has outperformed both two rules of Thumb and Arithmetic mean approaches. Secondly, we have adopted an efficient methodology to select an optimal set of the most relevant values of the parameters which are included in construction of BPNN classifier or impacting its performance. Those parameters are; the number of selected features/attributes, normalization of data, architecture of neural network specifically the number of nodes in the hidden layer, activation function, Learning rate and Momentum term. Except for Learning rate and Momentum term which are fixed at 0.01 value, for each parameter cited beforehand, we have chosen between two and four values that are used in other works and have yielded good results in term of intrusion detection. Searching of optimal values of both Learning Rate and Momentum is the focus of our current work. Our precedent approach was consisted of generating all possible combinations of these values (except for Learning rate and Momentum term) and then building the IDSs corresponding to each combination. Afterwards, we have compared the performance of the IDSs constructed by means of various performance measurements like accuracy, detection rate, F-score, false positive rate, AUC (ability to avoid false classification) etc., and selected the two best among them. Our preceding experiments were conducted on KDD CUP'99 and the results indicate that, compared to several traditional and recent techniques, the

proposed approach achieves higher detection rate, higher accuracy, higher F-score and lower false positive rate.
The best IDS obtained in our previous work have the characteristics shown in table 7.

5.3 The approach of our novel proposed system

In order to enhance the performance of our previous and best obtained ANIDS [32] based on BPNN classifier presented in table 3, we have implemented and integrated to that ANIDS a module of optimization based on Improved Genetic Algorithm (IGA). GA is improved through optimization strategies/techniques, namely Parallel Processing and Fitness Value Hashing, which decrease execution time, convergence time and save processing power (subsection 4.4). The purpose of developed module is searching the optimal values of Learning rate (LR) and Momentum term (MT) parameters, which influence the performance of BPNN classifier. In fact, LR and MT are two important parameters in the learning phase of BPNN, which impact the convergence and performance of that classifier. Since their values are between 0 and 1, we thought to use IGA in order to find the ideal/optimal values of those parameters in the interval [0; 1], which is a large space of real numbers.
As shown in figure 13, illustrated our approach, IGA process begins with a randomly generated population of individuals (potential solutions) represented by their chromosomes; each chromosome takes the form of a pair of values (Learning rate, Momentum). Then, this population evolves through several generations by means of genetic operations such elitism, selection, recombination (crossover) and mutation until stopping or optimization criterion of IGA is met. At each generation, each chromosome (chromosome = pair (LR, MT)) of current generation is evaluated by passing it as parameter to ANIDS obtained in our precedent work [32] , after using Min-Max normalization to convert the two substrings representing LR and MT into values between 0 and 1. This IDS firstly goes through the learning phase, then passes to the test/evaluation phase and returns the values of performance metrics calculated at the end of last phase. Among those performance metrics, we have selected the pertinent of them to serve as “Fitness Function” for evaluation of goodness of chromosomes. From one generation to the next, IGA converges towards the global optimum through genetic operations cited previously. Finally, the best individual (chromosome) is picked out as the final result once the optimization criterion is met. In our work, termination condition adopted for IGA is the maximum number of generations (100 generations). Hence, the best chromosome resulted correspond to the optimal or near-optimal values of the pair (Learning rate, Momentum), which ensuring high detection rate and low false alarm rate.
For successful use of IGA, two key elements must be well defined; the representation/encoding of chromosomes and the Fitness Function:

- **Chromosome encoding/representation:** In our study, we have chosen the binary representation for chromosomes. At the beginning of GA process; when the initial population of individuals was created, 50 binary strings of length 40 bits representing the chromosomes are generated randomly. Afterwards, each string is divided into two substrings of 20 bits. The first binary substring is then converted in decimal value, and this value is normalized using the Min Max normalization to get a value between 0 and 1and this value is normalized using the Min Max normalization to get a to get a value between 0 and 1, which will serve as the Learning rate of our IDS. Similarly, the foregoing approach is applied to the second binary substring to obtain Momentum term of our IDS, with a value within the interval [0; 1].
- **Fitness Function or Evaluation Function:** Among the values returned by the IDS at the end of the test/evaluation phase, we have chosen the AUC metric as a score (fitness function) of individuals to assess their adaptability to the optimization problem. The AUC [82] metric is by definition the ability to avoid misclassifications of network packets, and from our point of view, it represents a good compromise between the DR (Detection Rate) metric and the FPR (False Positive Rate) metric. In effect, this is due to the fact that AUC is the arithmetic mean of the DR and TNR (1-FPR) as shown by equation 2 of the AUC. As it is known, a good IDS is one that achieves a high detection rate (DR) and a low false positive rate (FPR). In fact, as the value of the DR measure increases and that of FPR measure decreases, consequently, the value of AUC increases. So, from our point of view, AUC is the best metric for evaluating an IDS. That is the reason of choice of AUC as fitness function.

$$AUC = \frac{(DR + TNR)}{2} = \frac{(DR + (1 - FPR))}{2} \quad (2)$$

In our optimization module based on IGA, we have employed the following algorithms/methods:

- Elitism;
- Roulette Wheel Selection;
- Single point Crossover;
- Bit flip mutation

5.4 The Framework of optimized ANIDS based on BPNN and IGA

Our optimized ANIDS (ANIDS BPNN-IGA) passes firstly through an optimization stage in order to be optimized. Consequently, it becomes ready to operate in operation/normal mode. The framework of our system ANIDS BPNN-IGA in optimization mode consists of four modules as illustrated in figure 5.

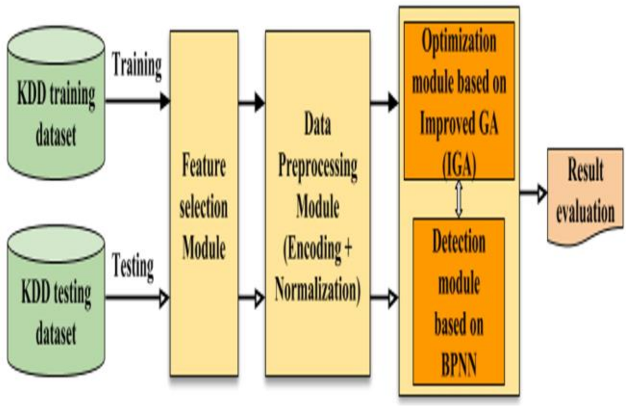

KDD training dataset
Training
KDD testing dataset
Testing
Feature selection Module
Data Preprocessing Module (Encoding + Normalization)
Optimization module based on Improved GA (IGA)
Detection module based on BPNN
Result evaluation

Figure 5. Framework of optimized ANIDS based on BPNN and IGA in optimization stage

- **Feature selection module:** Feature selection is the most critical stage in building intrusion detection models. Our intrusion detection model incorporates a feature selection module mainly to select useful features for intrusion detection. This module is based on A modified Kolmogorov–Smirnov Correlation Based Filter Algorithm which allows selection of a set of 12 relevant features among 41 features of KDD dataset.
- **Data preprocessing module:** Data Preprocessing includes two operations; data conversion (Categorical encoding) and Normalization. "Categorical encoding" refers to the process of assigning numeric values to nonnumeric features/attributes, so as to make the processing task much simpler, as numeric data can be easily handled upon. Whereas, "Normalization or Scaling" refers to the process of scaling the feature values to a small range that can help to obtain better detection results and avoid numerical difficulties during the calculation. Our data preprocessing module uses Min Max normalization method.
- **Detection module based on BPNN and optimization module based on IGA:** In order to enhance the performance of the detection module based on BPNN, this module interacts with an optimization module based on IGA as explained in details in subsection 5.3 with the goal to search the optimal values of Learning rate and Momentum term. The period of optimization is called "optimization stage". This period is achieved at the end of IGA process. Thus, the optimal values of Learning rate and Momentum term are found.

After passing through optimization phase and finding the optimal values of Learning rate and Momentum term, the optimized ANIDS (ANIDS BPNN-IGA) operates in operation/normal mode as shown in figure 6 to classify connection instances extracted from KDD '99 cup test dataset.

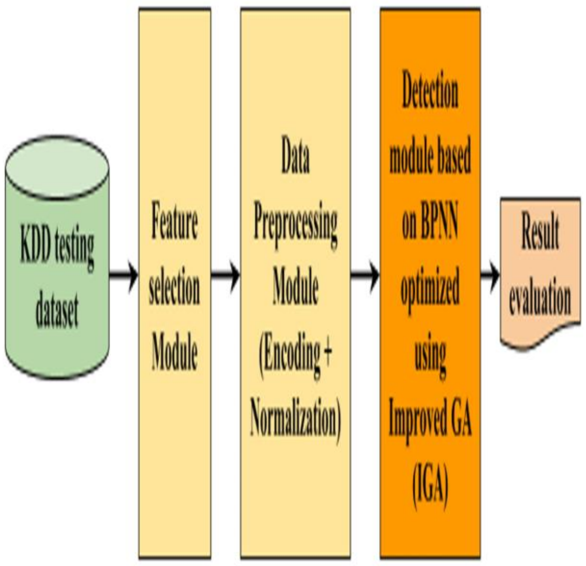

KDD testing dataset
Feature selection Module
Data Preprocessing Module (Encoding+ Normalization)
Detection module based on BPNN optimized using Improved GA (IGA)
Result evaluation

Figure 6. Framework of optimized ANIDS based on BPNN and IGA in operation/normal mode

6. Experimental Results and Analysis

This section is divided into three subsections, first one describes the performance measurements used for evaluation and comparison of original ANIDS based BPNN and optimized ANIDS based BPNN using IGA, the second gives an overview of our implementation and the third shows experimental results and analysis to select the best IDS . At end of last subsection, we have compared the performance of our new IDS to other related works.

6.1 Performance evaluation metrics

The ability of IDS to making the correct predictions, consider the measure of its effectiveness. Depending on the comparisons between the results that predict via intrusion detection system and the true nature of the event. There are four prospect outputs which are illustrated in table 4 known as confusion matrix. These four outcomes are true negative (TN) which indicates the correct prediction of normal behavior, true positive (TP) which indicates the correct predication of attack behavior, false positive (FP) which indicates the wrong predication of normal behavior as attack and false negative (FN) which indicates mistake predication of attack behavior as normal. Both (TN) and (TP) are considered guide of the correct operation of the IDS. Besides, FN and FP rates reduce the effectiveness of IDS where FP reduce the capability of system in detection and FN will make the system susceptible to intrusion [10, 15]. Therefore, for an IDS to be effective, the FP and FN rates should be minimized, and TP and TN rates to be maximized.

Table 1. Confusion matrix

Actual class	Predicted class	
	Attack	**Normal**
Attack	True positive (TP)	False negative(FP)
Normal	False positive (FP)	True negative(TN)

- **True Positive (TP):** Number of instances correctly predicted as attacks.
- **False Positive (FP):** Number of instances wrongly predicted as attacks.
- **True Negative (TN):** Number of instances correctly predicted as non-attacks (normal instances).
- **False Negative (FN):** Number of instances wrongly predicted as non-attacks [1].

Based on confusion matrix shown in table 1 [12], to determine the performance of IDSs the following metrics used for the numerical evaluation. These performance criteria are not dependent on the size of the training and testing samples and can be really helpful in assessing the performance of the ensemble model [38, 83]:

- **Accuracy (ACC):** Is the ratio of samples which are correctly predicted as normal or attack to the overall number of samples in test set. Overall, it measures how often the classifier is correct and is calculated using equation 3.

$$Accuracy(ACC) = \frac{TP + TN}{TP + TN + FP + FN} \quad (3)$$

- **Detection Rate (DR)/Recall:** Called also True Positive Rate (TPR) or Sensitivity. It is a performance metric which indicates the ratio of the number of samples that are correctly classified as attack to the total number of attack samples present in test set and is calculated using equation 4.

$$Detection\ Rate\ (\mathrm{DR/Recall}) = \frac{TP}{TP + FN} \quad (4)$$

- **Precision:** It indicates the percentage of intrusions that have occurred, and the IDS detects them correctly. It is calculated by the number of correctly classified positive (intrusion) examples divided by the number of examples labeled by the system as positive. It is calculated using equation 5.

$$\mathrm{Precision} = \frac{TP}{TP + FP} \quad (5)$$

- **True Negative Rate (TNR):** Called also Specificity. Is the ratio of the number of legitimate records detected as normal instances divided by total number of normal (legitimate) instances included in the test set. It is calculated using equation 6.

$$True\ Negative\ Rate\ (\mathrm{TNR}) = \frac{TN}{TN + FP} \quad (6)$$

- **False Alarm Rate (FAR):** Also known as False Positive Rate (FPR) refers to the proportion of normal packets being falsely detected as malicious. In other way, it represents the ratio of samples which is improperly categorized as attack to the overall number of samples of normal behavior. If this value is consistently elevated, it causes the administrator to intentionally disregard the system warnings, which makes the system enter into a dangerous status. Thus, it should be as minimum as possible. The FPR value can be calculated by one minus TNR. FAR is obtained from equation 7.

$$False\ Alarm\ Rate\ (FAR) = \frac{FP}{FP + TN} \quad (7)$$

- **False Negative Rate (FNR):** Is the ratio of the number of attack instances detected as normal instances divided by total attack instances included in the test set. This term is used to describe a network intrusion device's inability to detect the true attack. The FNR value can be calculated by one minus TPR. FNR is obtained from equation 8.

$$False\ Negative\ Rate\ (FNR) = \frac{FN}{FN + TN} \quad (8)$$

- **F-Score:** Also called as F-measure. This is a statistical criterion to evaluate the performance of the systems. In fact, this criterion is a harmonic mean between metrics recall (true positive rate) and precision, which includes a range of values between 0 and 1. The higher value of F-score indicates that the IDS is performing better on recall and precision. F-Score is obtained from the equation 9 [84].

$$F - score = \frac{2 \times Re\,call \times Precision}{Re\,call + Pr\,ecision} \quad (9)$$

- **AUC:** Is ability to avoid false classification [82] and is calculated using equation 10.

$$AUC = 0.5 \times \left(\frac{TP}{TP+FN} + \frac{TN}{TN+FP}\right) \quad (10)$$

- **The average time to classify a connection instance (AVGTC):** The average time of classification of a connection instance is defined as the total time of the classification of all connection instances included in the test set divided by the test set size or the number of connection instances present in test set. This is a substantial evaluation criterion to evaluate the efficiency of the IDS. Because, if this time is high, in on one hand, attackers can detect the presence of the IDS, and then they will be a target of attacks lunched against the IDS itself to paralyze it. On the other hand, that can cause packet loss [32].

6.2 Implementation

For experimental set up, we have used a computer with a Core-i7 2700K CPU and 16 GB of DDR3. For simulation, we have employed Cloudsim 4.0 and DARPA's cup datasets 1999. Our proposed system is developed using Java language. Below, table 2 presents the parameters of our module of optimization based on IGA developed for our previous ANIDS [32].

Table 2. Parameters of our module of optimization based on Improved Genetic Algorithm for ANIDS BPNN

Parameters of Improved Genetic Algorithm	Value
Length of chromosomes	40 bits
Elitism number: the number of best chromosomes which will be copied without changes to a new population (next generation)	5
Population size	50
Maximum number of generations	100
Crossover rate	0.95
Mutation rate	0.1

6.3 Evaluation and analysis

Like many research [85, 44], in our experiments, we have used two independent KDD cup 99 datasets, commonly employed and are available in [86]. The first one is used for training the IDSs and the second is used for testing or evaluating these IDSs. The KDD training dataset is **kddcup.data_10_percent.gz 10% dataset**, from which we have extracted the training dataset for our IDSs (IDS based BPNN and IDS based BPNN-IGA) and the KDD testing dataset is **corrected.gz KDD**, which is used to build our testing dataset. Table 3 shows the details of our datasets.

Table 3. Distribution and size of train and test datasets

Class of records	Corrected KDD 99 (training dataset)	10% KDD (testing dataset)
Normal	15000	20000
Dos	60000	59245
Probe	2490	585
R2L	165	170
U2R	30	260
Total of records	**77685**	**80260**

The experiments conducted on our proposed system show that at the end of IGA process that is to say after 100 generations, the best individual (chromosome) generated by IGA, contains the value ***9.536752259018191E-7*** for Learning rate and the value ***1.649858140810147E-4*** for Momentum term, and this chromosome or this pair of values (Learning rate, Momentum term) allows our optimized ANIDS to reach the value *99.18%* for *AUC*, which is the performance metric for the ANIDS adopted as Fitness function. In addition, in the same time, our optimized ANIDS have achieved *detection rate (DR)* value of *98.46%, Precision* of *99.96%* and False *Positive Rate (FPR)* of *0.11%*. Thus, the optimal or near-optimal values of Learning rate and Momentum term are *9.536752259018191E-7* and *1.649858140810147E-4* respectively, which ensure high performance of our optimized ANIDS.As shown in table 4, our optimized ANIDS BPNN by using IGA outperforms the original ANIDS BPNN developed in our previous work

Table 4. Comparaion of performance between our IDSs

Performance Metrics	Original ANIDS based on BPNN	Proposed IDS ANID BPNN-IGA
Accuracy	98.66%	98.82%
Precision	99.62%	99.96%
FPR	1.13%	0.11%
FNR	1.41%	1.54%
TPR (DR)	98.59%	98.46%
TNR	98.87%	99.89%
F-score	0.99	0.99
AUC	98.73%	99.18%
Average of time of classification	0.0000890854 s	0.0000897584 s

We have compared the performance of the two IDSs; original ANIDS BPNN and optimized ANIDS BPNN-IGA using various performance measurements as shown below.

Figure 7 shows that original ANIDS BPNN and optimized ANIDS BPNN-IGA have almost the same detection rate; the first one attains the value of 98.59% while the second attains the value of 98.46%. However, ANIDS BPNN-IGA outperforms ANIDS BPNN in terms of TNR (True Negative Rate); they reach values of 99.89% and 98.87% respectively. So, ANIDS BPNN-IGA has the ability to classify normal connection instances better than ANIDS BPNN. In fact, figure 08 confirms this ability; ANIDS BPNN-IGA presents lower FPR (0.11%) than FPR (1.13%) of ANIDS BPNN.

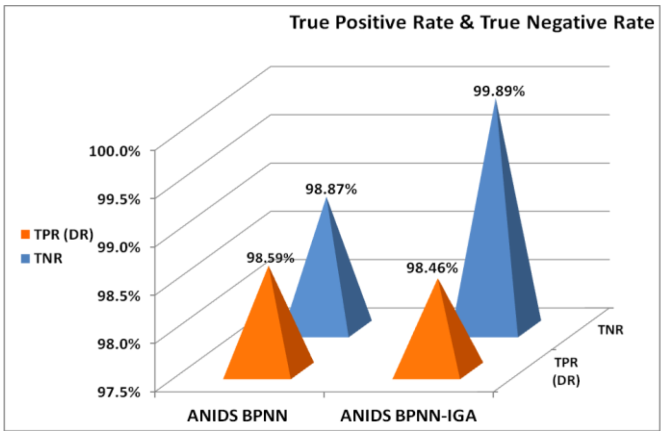


Figure 7. True Positive Rate (DR) and True Negative Rate for ANIDS BPNN and ANIDS BPNN-IGA

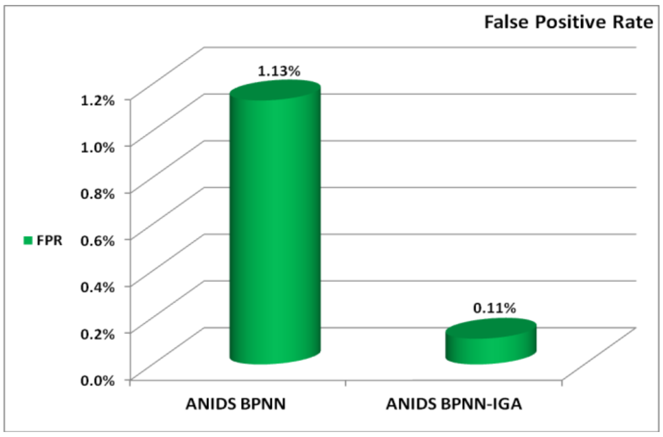


Figure 8. False Positive Rate (FPR) for ANIDS BPNN and ANIDS BPNN-IGA

Figure 9 indicates that ANIDS BPNN-IGA has the ability to avoid false classifications better than ANIDS BPNN; the first one reaches the value of 99.18% for AUC, while the second has the value of 98.73% for AUC.

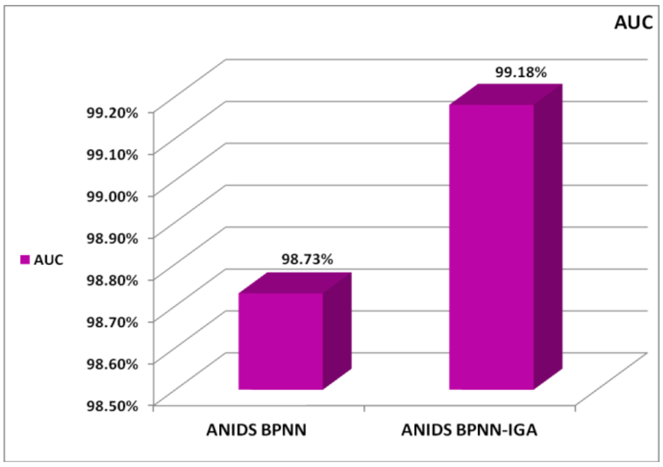


Figure 9. Ability to avoid false classification (AUC) for ANIDS BPNN and ANIDS BPNN-IGA

Figure 10 displays that both ANIDS BPNN and ANIDS BPNN-IGA yield the same value of F-score which is 0.99.

The high value of F-score (0.99) demonstrates that those IDSs are performing better on recall and precision.

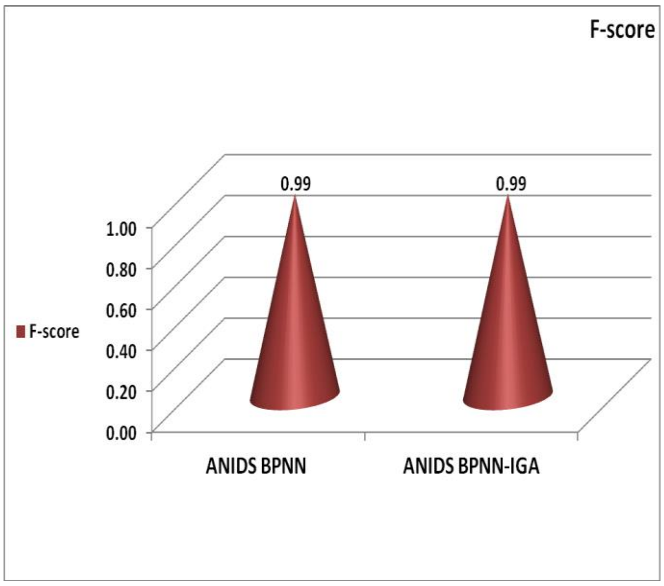


Figure 10. F-score for ANIDS BPNN and ANIDS BPNN-IGA

As shown in figure 11, ANIDS BPNN-IGA outperforms ANIDS in terms of accuracy and precision. Concerning Accuracy metric, ANIDS BPNN-IGA attains the value of 98.82% while ANIDS BPNN attains the value of 98.66%.
For Precision metric, ANIDS BPNN-IGA reaches the value of 99.96% whereas ANIDS BPNN reaches the value of 98.62%.

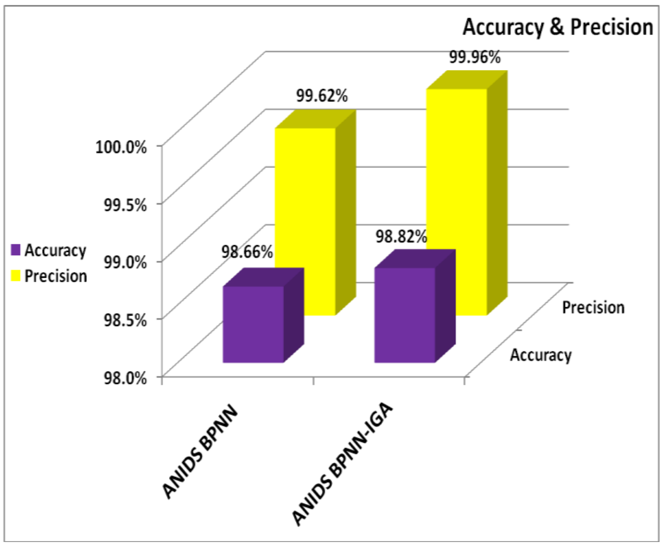


Figure 11. Accuracy and Precision for ANIDS BPNN & ANIDS BPNN-IGA

After considering all performance measurements used in our experiments, we have come to conclusion that our proposed ANIDS BPNN-IGA, which is our previous ANIDS based on BPNN and optimized using Improved Genetic Algorithm outperforms the original ANIDS BPNN.
As shown in table 8, the comparison to several works indicates that our proposed approach achieves higher detection rate and lower false positive rate.
Further, using of IGA, obtained by application of optimization strategies to the fitness function of a standard GA, namely Parallel Processing and Fitness Value Hashing have brought several benefits. These advantages are: 85% reduction of execution time compared to a standard GA, acceleration of the IGA convergence process and save of processing power.

7. Conclusions

In order to enhance the performance of our previous ANIDS based BPNN, we have developed for it a module of optimization based on Improved GA (IGA) with the purpose of searching the optimal values of critical parameters for BPNN, namely Learning rate (LR) and Momentum term (MT). GA was improved through optimization strategies that is Parallel Processing and Fitness Value Hashing. We have used binary encoding for chromosomes; where the first half part represents LR, whereas the second half part represents MT. Further, we have adopted AUC metric as fitness function of IGA. Each chromosome generated by IGA is introduced to the ANIDS, which thereafter goes through learning phase and a test phase. At the end of the last phase, AUC measure is computed. Finally, at the end IGA process, the best value of LR and MT are found. Experimental results conducted using CloudSim 4.0 and KDD CUP' 99 datasets demonstrate that our IDS "ANIDS BPNN-IGA" outperforms original ANIDS BPNN and several recent works. In addition, performance improvement strategies applied to GA have reduced execution time, convergence time and saved processing power. We have chosen to place the proposed system "ANIDS BPNN-IGA" on Front-End and Back-End of the Cloud, to detect and stop attacks in real time impairing the security of the Cloud Datacenter. Our future work is to combine our new IDS with Snort our Suricata IDS to build a more efficient and effective hybrid IDS.

References

[1] A. Uzma, K.K. Dewangan, D.K. Dewangan, "Distributed Denial of Service Attack Detection Using Ant Bee Colony and Artificial Neural Network in Cloud Computing," in: B. Panigrahi, M. Hoda, V. Sharma, S. Goel (Eds.), Nature Inspired Computing, Advances in Intelligent Systems and Computing, Vol. 652, Springer, Singapore, Singapore, pp. 165-175, 2018.
[2] C. Modi, D. Patel, B. Borisaniya, H. Patel, A. Patel, M. Rajarajan, "A survey of intrusion detection techniques in cloud," Journal of Network and Computer Applications, Vol. 36, No. 1, pp. 42-57. 2013.
[3] X. Zhao, W. Zhang, "Hybrid Intrusion Detection Method Based on Improved Bisecting K-Means in Cloud Computing," 13th IEEE Web Information Systems and Applications Conference (WISA), Wuhan, China, pp. 225-230, 2016.
[4] G. Brunette, R. Mogull, "Security guidance for critical areas of focus in cloud computing v2. 1," Cloud Security Alliance, pp. 1-76, 2009.
[5] Q. Zhang, L. Cheng, R. Boutaba, "Cloud computing: state-of-the-art and research challenges," Journal of internet services and applications, Vol. 1, No. 1, pp. 7-18, 2010.
[6] M. A. Hatef, V. Shaker, M. R. Jabbarpour, J. Jung, H. Zarrabi, "HIDCC: A hybrid intrusion detection approach in cloud computing," Concurrency and Computation: Practice and Experience, Vol. 30, No. 3, 2018.
[7] M. Ali, S. U. Khan, A. V. Vasilakos, "Security in cloud computing: Opportunities and challenges," Information sciences, Vol. 305, pp. 357-383, 2015.
[8] A. S. Saljoughi, M. Mehrvarz, H. Mirvaziri, "Attacks and Intrusion Detection in Cloud Computing Using Neural

Networks and Particle Swarm Optimization Algorithms," Emerging Science Journal, Vol. 1, No. 4, pp. 179-191, 2018.
[9] V. Marinova-Boncheva, "A short survey of intrusion detection systems," problems of Engineering Cybernetics and Robotics, Vol. 58, pp. 23-30, 2007.
[10] S. M. Mehibs, S. H. Hashim, "Proposed Network Intrusion Detection System Based on Fuzzy c Mean Algorithm in Cloud Computing Environment," Journal of University of Babylon, Vol. 26, No. 2, pp. 27-35, 2018.
[11] A. Patel, M. Taghavi, K. Bakhtiyari, J. C. JúNior, "An intrusion detection and prevention system in cloud computing: A systematic review," Journal of network and computer applications, Vol. 36, No. 1, pp. 25-41, 2013.
[12] D. A. Kumar, S. R. Venugopalan, " A Novel Algorithm for Network Anomaly Detection Using Adaptive Machine Learning," In: K. Saeed, N. Chaki, B. Pati, S. Bakshi, D. Mohapatra (Eds.), Progress in Advanced Computing and Intelligent Engineering, Advances in Intelligent Systems and Computing, Vol. 564, Springer, Singapore, Singapore, pp. 59-69, 2018.
[13] K. Scarfone, P. Mell, "Guide to intrusion detection and prevention systems (idps) ," NIST special publication, Vol. 800, No. 2007, p.94, 2007.
[14] Z. Chiba, N. Abghour, K. Moussaid, M. Rida, "A cooperative and hybrid network intrusion detection framework in cloud computing based on snort and optimized back propagation neural network," Procedia Computer Science, Vol. 83, pp. 1200-1206, 2016.
[15] S. M. Mehibs, S. H. Hashim, "Proposed Network Intrusion Detection System In Cloud Environment Based on Back Propagation Neural Network," Journal of University of Babylon for Pure and Applied Sciences, Vol. 26, No. 1, pp. 29-40, 2018.
[16] W. Yassin, N. I. Udzir, Z. Muda, A. Abdullah, M. T. Abdullah, "A cloud-based intrusion detection service framework", 2012 International Conference on Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), Kuala Lumpur, Malaysia, pp. 213-218, 2012.
[17] S. X. Wu, W. Banzhaf, "The use of computational intelligence in intrusion detection systems: A review", Applied Soft Computing, Vol. 10, No. 1, pp. 1-35, 2010.
[18] B. Shah, B. H. Trivedi, "Artificial neural network based intrusion detection system: A survey", International Journal of Computer Applications, Vol. 39, No. 6, pp. 13-18, 2012.
[19] S. Borah, R. Panigrahi, A. Chakraborty, "An Enhanced Intrusion Detection System Based on Clustering," in: K. Saeed, N. Chaki, B. Pati, S. Bakshi, D. Mohapatra (Eds.), Progress in Advanced Computing and Intelligent Engineering, Advances in Intelligent Systems and Computing, Vol. 564, Springer, Singapore, Singapore, pp. 37-45, 2018.
[20] S. A. Aljawarneh, R. A. Moftah, A. M. Maatuk, "Investigations of automatic methods for detecting the polymorphic worms signatures", Future Generation Computer Systems, Vol. 60, pp. 67-77, 2016.
[21] Z. Chiba, N. Abghour, K. Moussaid, M. Rida, "A Review of Intrusion Detection Systems in Cloud Computing," Security and Privacy in Smart Sensor Networks, pp. 253-283, 2018, IGI Global.
[22] S. G. Kene, D. P. Theng, "A review on intrusion detection techniques for cloud computing and security challenges", 2015 2nd International Conference on Electronics and Communication Systems (ICECS), Coimbatore, India, pp. 227-232, 2015.
[23] Munir, K., Palaniappan, S. (2012). Security threats/attacks present in cloud environment. IJCSNS, 12(12), 107. pp. 107, 2012.
[24] N. Pandeeswari, G. Kumar, "Anomaly detection system in cloud environment using fuzzy clustering based ANN", Mobile Networks and Applications, Vol. 21, No. 3, pp. 494-505, 2016.
[25] H. Debar, M. Dacier, A. Wespi, "A revised taxonomy for intrusion-detection systems", Annales des telecommunications, Vol. 55, No. 7-8, pp. 361-378, 2000. Springer-Verlag.
[26] U. Oktay and O. K. Sahingoz, "Proxy network intrusion detection system for cloud computing," 2013 IEEE International Conference on Technological Advances in Electrical, Electronics and Computer Engineering (TAEECE), Konya, Turkey, pp. 98-104, 2013.
[27] C. Kolias, G. Kambourakis and M. Maragoudakis, "Swarm intelligence in intrusion detection: A survey," Computers & Security, vol. 30, no. 8, pp. 625-642, 2011.
[28] M. El Boujnouni, M. Jedra, " New intrusion detection system based on support vector domain description with information gain metric," International Journal of Network Security, Vol. 20, No. 1, pp. 25-34, 2018.
[29] J. Choi, C. Choi, B. Ko, P. Kim, " A method of DDoS attack detection using HTTP packet pattern and rule engine in cloud computing environment," Soft Computing, Vol. 18, No. 9, pp. 1697-1703, 2014.
[30] P. Ghosh, S. Jha, R. Dutta, S. Phadikar, "Intrusion Detection System Based on BCS-GA in Cloud Environment," in: N. Shetty, L. Patnaik, N. Prasad, N. Nalini (Eds.), Emerging Research in Computing, Information, Communication and Applications (ERCICA 2016), Springer, Singapore, Singapore, pp. 393-403, 2018.
[31] Y. Mehmood, M. A. Shibli, U. Habiba, R. Masood, "Intrusion detection system in cloud computing: challenges and opportunities," 2013 2nd National Conference on Information Assurance (NCIA), Rawalpindi, Pakistan, pp. 59-66, 2013.
[32] Z. Chiba, N. Abghour, K. Moussaid, M. Rida, "A novel architecture combined with optimal parameters for back propagation neural networks applied to anomaly network intrusion detection," Computers & Security, Vol. 75, pp. 36-58, 2018.
[33] E. U. Opara, O. A. Soluade, " Straddling the next cyber frontier: The empirical analysis on network security, exploits, and vulnerabilities," International Journal of Electronics and Information Engineering, Vol. 3, No. 1, pp. 10-18, 2018.
[34] A. Tayal, N. Mishra, S. Sharma, "Active monitoring & postmortem forensic analysis of network threats: A survey," International Journal of Electronics and Information Engineering, Vol. 6, No. 1, pp. 49-59, 2017.
[35] K. Vieira, A. Schulter, C. Westphall, C. Westphall, "Intrusion detection for grid and cloud computing," It Professional, Vol. 12, No. 4, pp. 38-43, 2010.
[36] R. Bace, P. Mell, "Special Publication on Intrusion Detection Systems," Tech. Report SP 800-31, National Institute of Standards and Technology, Gaithersburg, Md., 2001.
[37] M.A. Salama, H.F. Eid, R.A. Ramadan, A. Darwish, A.E. Hassanien, "Hybrid Intelligent Intrusion Detection Scheme," In: A. Gaspar-Cunha, R. Takahashi, G. Schaefer, L. Costa (Eds.), Soft Computing in Industrial Applications, Advances in Intelligent and Soft Computing, Vol. 96, Springer, Berlin, Heidelberg, Germany, pp. 293.-303, 2011.
[38] M. Amini, J. Rezaeenour, E. Hadavandi, " A neural network ensemble classifier for effective intrusion detection using fuzzy clustering and radial basis function networks," International Journal on Artificial Intelligence Tools, Vol. 25, No. 02, pp. 1550033-1550062, 2016.
[39] A.A. Mohd, P. Angelov, "Anomalous Behaviour Detection Based on Heterogeneous Data and Data Fusion," Soft Computing, Vol. 22, No. 10, pp. 3187-3201, 2017.
[40] Z. Chiba, N. Abghour, K. Moussaid, A. El Omri and M. Rida, "A survey of intrusion detection systems for cloud computing environment", IEEE International Conference on Engineering & MIS (ICEMIS),Madrid, Spain, pp. 1-13, 2016.

[41] J. Z. Zhao, H. K. Huang, "An intrusion detection system based on data mining and immune principles", International Conference on Machine Learning and Cybernetics, Beijing, China, pp. 524-528, 2002.

[42] W. Yurcik, "Controlling intrusion detection systems by generating false positives: squealing proof-of-concept", 27th Annual IEEE Conference on Local Computer Networks, Tampa, FL, USA, pp. 134-135, 2002.

[43] C. N. Modi, D. R. Patel, A. Patel, M. Rajarajan, " Integrating signature apriori based network intrusion detection system (NIDS) in cloud computing," Procedia Technology, Vol. 6, pp. 905-912, 2012.

[44] N. Modi and D. Patel, "A novel hybrid-network intrusion detection system (H-NIDS) in cloud computing", 2013 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), Singapore, Singapore, pp. 23-30,2013.

[45] M. Moorthy, M. Rajeswari, "Virtual host based intrusion detection system for cloud," Journal of Engineering & Technology, pp. 0975-4024, 2013.

[46] Y. Mehmood, M. A. Shibli, A. Kanwal, R. Masood, "Distributed intrusion detection system using mobile agents in cloud computing environment," 2015 Conference on Information Assurance and Cyber Security (CIACS), Rawalpindi, Pakistan, pp. 1-8, 2015.

[47] S. Sangeetha, B. Gayathri devi, R. Ramya, M.K. Dharani, P. Sathya, " Signature Based Semantic Intrusion Detection System on Cloud," In: J. Mandal, S. Satapathy, M. Kumar Sanyal, P. Sarkar, A. Mukhopadhyay (Eds.), Information Systems Design and Intelligent Applications, Advances in Intelligent Systems and Computing, Vol. 339. Springer, New Delhi, India, pp. 657-666, 2015.

[48] D. Singh, D. Patel, B. Borisaniya, C. Modi, "Collaborative ids framework for cloud," Journal of Network Security, Vol. 18, No. 4, pp. 699-709, 2016.

[49] Q. Qian, J. Cai, R. Zhang, " Intrusion detection based on neural networks and artificial bee colony algorithm," 2014 IEEE/ACIS 13th International Conference on Computer and Information Science (ICIS), Taiyuan, China, pp. 257-262, 2014.

[50] J. B. Li, Y. K. Chung, " A novel back-propagation neural network training algorithm designed by an ant colony optimization," 2005 IEEE/PES Transmission & Distribution Conference & Exposition: Asia and Pacific, Dalian, China, pp. 1-5, 2005.

[51] L. B. Lai, R. I. Chang, J. S. Kouh, "Detecting network intrusions using signal processing with query-based sampling filter," EURASIP Journal on Advances in Signal Processing, Vol. 2009, No. 1, pp. 1-8, 2008.

[52] G. P. Zhang, "Neural networks for classification: a survey," IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), Vol. 30, No. 4, pp. 451-462, 2000.

[53] N. Sen, R. Sen, M. Chattopadhyay, "An effective back propagation neural network architecture for the development of an efficient anomaly based intrusion detection system," 2014 International Conference on Computational Intelligence and Communication Networks, Bhopal, India, pp. 1052-1056, 2014.

[54] G. Song, J. Zhang and Z. Sun, "The research of dynamic change learning rate strategy in BP neural network and application in network intrusion detection", IEEE 3rd International Conference on Innovative Computing Information and Control, 2008. ICICIC'08, Dalian, Liaoning, China, pp. 513-513, 2008.

[55] C. Qiu, J. Shan, B. Shandong, "Research on intrusion detection algorithm based on BP neural network," International Journal of Security and its Applications, Vol. 9, No. 4, pp. 247-258, 2015.

[56] C. Chang, X. Sun, D. Chen, C. Wang, "Application of Back Propagation Neural Network with Simulated Annealing Algorithm in Network Intrusion Detection Systems," In: S. Sun, N. Chen, T. Tian (Eds.), Signal and Information Processing, Networking and Computers, ICSINC 2017, Lecture Notes in Electrical Engineering, Vol. 473. Springer, Singapore, Singapore, pp. 172-180, 2018.

[57] S. Haykin, "Neural Networks: A Comprehensive Foundation", 2nd Edition, Pearson, 1999.

[58] Multi-Layer Perceptron, 2018. http://www.cse.unsw.edu.au/~cs9417ml/MLP2/

[59] B. Uppalaiah, K. Anand, B. Narsimha, S. Swaraj, T. Bharat, "Genetic Algorithm Approach to Intrusion Detection System," IJCST, Vol. 3, No. 1, pp. 156-160 , 2012.

[60] S.M. Bridges, R.B. Vaughn, "Fuzzy data mining and genetic algorithms applied to intrusion detection," Twenty third National Information Security Conference, Baltimore, MD, USA, pp. 13-31, 2000.

[61] V. R. Chaudhary, R. S. Bichkar, "Detection of Intrusions in KDDCup Dataset using GA by Enumeration Technique," International Journal of Innovative Research in Computer and Communication Engineering, Vol. 3, No. 3, pp. 2365-2369, 2015.

[62] P. S. Maniyar, V. Musande, "Rules Based Intrusion Detection System Using Genetic Algorithm," International Journal of Computer Science and Network, Vol. 5, No. 3, pp. 554-558, 2016.

[63] A. S. A. Aziz, M. A. Salama, A. ella Hassanien, S. E. O. Hanafi, "Artificial immune system inspired intrusion detection system using genetic algorithm," Informatica, Vol. 36, No. 4, pp. 347–357, 2012.

[64] S. Selvakani, R. S. Rajesh, "Genetic Algorithm for framing rules for Intrusion Detection," International Journal of Computer Science and Network Security, Vol. 7, No. 11, pp. 285-290, 2007.

[65] L. Jacobson, B. Kanbe, "Genetic algorithms in Java basics," Apress, New York, USA, 2015.

[66] V. Shah, A. K. Aggarwal, N. Chaubey, "Performance improvement of intrusion detection with fusion of multiple sensors," Complex & Intelligent Systems, Vol. 3, No. 1, pp. 33-39, 2017.

[67] P. Ghosh, C. Debnath, D. Metia, D. R. Dutta, "An efficient hybrid multilevel intrusion detection system in cloud environment," IOSR Journal of Computer Engineering (IOSR-JCE), Vol. 16, No. 4, pp. 16-26, 2014.

[68] P. Chouhan, V. Richhariya, "Anomaly Detection in Network using Genetic Algorithm and Support Vector Machine," International Journal of Computer Science and Information Technologies (IJCSIT), Vol. 6, No. 5, pp. 4429-4433, 2015.

[69] J. Frank, "Artificial intelligence and intrusion detection: Current and future directions," 17th national computer security conference, Baltimore, MD, USA, pp. 1-12, 1994.

[70] W. Lee, S. J. Stolfo, "A framework for constructing features and models for intrusion detection systems," ACM transactions on Information and system security (TiSSEC), Vol. 3, No. 4, pp. 227-261, 2000.

[71] V. R. Balasaraswathi, M. Sugumaran, Y. Hamid, "Feature selection techniques for intrusion detection using non-bio-inspired and bio-inspired optimization algorithms," Journal of Communications and Information Networks, Vol. 2, No. 4, pp. 107-119, 2017.

[72] M. A. Ambusaidi, X. He, P. Nanda, Z. Tan, "Building an intrusion detection system using a filter-based feature selection algorithm," IEEE transactions on computers, Vol. 65, No. 10, pp. 2986-2998, 2016.

[73] Z. A. Baig, A. S. Shaheen, R. AbdelAal, "One-dependence estimators for accurate detection of anomalous network

traffic," International Journal for Information Security Research (IJISR), Vol. 1, No. 4, pp. 202-210, 2011.
[74] P. Mzila, E. Dube, "The effect of destination linked feature selection in real-time network intrusion detection," Eighth International Conference on Internet Monitoring and Protection, Rome, Italy, pp. 8-13, 2013.
[75] A. Gomathy, B. Lakshmipathi, "Network Intrusion Detection Using Genetic Algorithm and Neural Network," In: D.C. Wyld, M. Wozniak, N. Chaki, N. Meghanathan, D. Nagamalai (Eds.), Advances in Computing and Information Technology (ACITY 2011), Communications in Computer and Information Science, Vol. 198. Springer, Berlin, Heidelberg, Germany, pp. 399-408, 2011.
[76] K. Subrahmanyam, N. S. Sankar, S. P. Baggam, R. S. Raghavendra, "A Modified KS-test for Feature Selection," IOSR Journal of Computer Engineering (IOSR-JCE), Vol. 13, No. 3, pp. 73-79, 2013.
[77] K. M. Ali, V. W. Samawi, M. S. Al Rababaa, "The affect of fuzzification on neural networks intrusion detection system," 2009 4th IEEE Conference on Industrial Electronics and Applications, Xi'an, China, pp. 1236-1241, 2009.
[78] Z. Ihsan, M. Y. Idris, A. H. Abdullah, "Attribute normalization techniques and performance of intrusion classifiers: A comparative analysis," Life Science Journal, Vol. 10, No. 4, pp. 2568-2576, 2013.
[79] N. Lokeswari, B. C. Rao, "Artificial Neural Network Classifier for Intrusion Detection System in Computer Network", Proceedings of the Second International Conference on Computer and Communication Technologies. Hyderabad, India, pp. 581-591, 2016
[80] M. Panda, M. R. Patra, " Network intrusion detection using naive bayes," International journal of computer science and network security, Vol. 7, No. 12, pp. 258-263, 2007.
[81] C. N. Modi, D. R. Patel, A. Patel, R. Muttukrishnan, "Bayesian Classifier and Snort based network intrusion detection system in cloud computing," 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), Coimbatore, India, pp. 1-7, 2012.
[82] M. Sokolova, G. Lapalme, "A systematic analysis of performance measures for classification tasks," Information Processing & Management, Vol. 45, No. 4, pp. 427-437, 2009.
[83] E. Chakir, I. K. Youness, M. Moughit, "False positives reduction in intrusion detection systems using Alert correlation and datamining techniques," Int J Adv Res Comput Sci Softw Eng, Vol. 5, No. 4, pp. 77-85, 2015.
[84] M. H. Aghdam, P. Kabiri, "Feature Selection for Intrusion Detection System Using Ant Colony Optimization," IJ Network Security, Vol. 18, No. 3, pp. 420-432, 2016.
[85] S. Kumar, A. Yadav, "Increasing performance Of intrusion detection system using neural network," 2014 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Ramanathapuram, India, pp. 546-550, 2014.
[86] KDD Cup 1999 data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.
[87] R. Gaidhane, C. Vaidya, M. Raghuwanshi, "Intrusion Detection and Attack Classification using Back-propagation Neural Network," International Journal of Engineering Research and Technology (IJERT), Vol. 3, No. 3, pp. 1112-1115, 2014.
[88] B. Al-Shdaifat, W. S. Alsharafat, M. El-bashir, "Applying Hopfield Artificial Network and Simulating Annealing for Cloud Intrusion Detection, " Journal of Information Security Research, Vol. 6, No. 2, pp. 49-53, 2015.
[89] P. Ghosh, A. K. Mandal, R. Kumar, "An Efficient Cloud Network Intrusion Detection System," in: J. Mandal, S. Satapathy, M. Kumar Sanyal, P. Sarkar, A. Mukhopadhyay (Eds.), Information Systems Design and Intelligent Applications, Advances in Intelligent Systems and Computing, Vol. 339, Springer, New Delhi, India, pp. 91-99, 2015.
[90] W. L. Al-Yaseen, Z. A. Othman, M. Z. A. Nazri, "Intrusion detection system based on modified k-means and multi-level support vector machines," International Conference on Soft Computing in Data Science, Putrajaya, Malaysia ,pp. 265-274, 2015.
[91] M. Zhang, B. Xu, D. Wang, "An Anomaly Detection Model for Network Intrusions Using One-Class SVM and Scaling Strategy," International Conference on Collaborative Computing: Networking, Applications and Worksharing. Wuhan, China, pp. 267-278, 2015.
[92] B. M. Aslahi-Shahri, R. Rahmani, M. Chizari, A. Maralani, M. Eslami, M. J. Golkar, A. Ebrahimi, "A hybrid method consisting of GA and SVM for intrusion detection system," Neural computing and applications, Vol. 27, No. 6, pp. 1669-1676, 2016.
[93] S. Sharma, A. Gupta, S. Agrawal, "An Intrusion Detection System for Detecting Denial-of-Service Attack in Cloud Using Artificial Bee Colony," In: S. Satapathy, Y. Bhatt, A. Joshi, D. Mishra (Eds.), Proceedings of the International Congress on Information and Communication Technology, Advances in Intelligent Systems and Computing, Vol. 438. Springer, Singapore, Singapore, pp. 137-145, 2016.
[94] N. Belhadj-Aissa, M. Guerroumi, "A New Classification Process for Network Anomaly Detection Based on Negative Selection Mechanism," Security, Privacy and Anonymity in Computation, Communication and Storage: SpaCCS 2016 International Workshops, TrustData, TSP, NOPE, DependSys, BigDataSPT, and WCSSC, Zhangjiajie, China, pp. 238-248, 2016.
[95] M. E. Aminanto, K. I. M. HakJu, K. I. M. Kyung-Min, K. I. M. Kwangjo, "Another Fuzzy Anomaly Detection System Based on Ant Clustering Algorithm," IEICE TRANSACTIONS on Fundamentals of Electronics, Communications and Computer Sciences, Vol. E100-A, No. 1, pp. 176-183, 2017.
[96] H. H. Pajouh, G. Dastghaibyfard, S. Hashemi, "Two-tier network anomaly detection model: a machine learning approach", Journal of Intelligent Information Systems, Vol. 48, No.1, pp. 1-14, 2015.
[97] T. Omrani, A. Dallali, B. C. Rhaimi, J. Fattahi, " Fusion of ANN and SVM classifiers for network attack detection," 2017 18th International Conference on Sciences and Techniques of Automatic Control and Computer Engineering (STA), Monastir, Tunisia, pp. 374-377, 2017.
[98] T. Ma, Y. Yu, F. Wang, Q. Zhang, X. Chen, "A Hybrid Methodologies for Intrusion Detection Based Deep Neural Network with Support Vector Machine and Clustering Technique," In: N. Yen, J. Hung (Eds.), Frontier Computing (FC 2016), Lecture Notes in Electrical Engineering, Vol. 422, Springer, Singapore, Singapore, pp. 123-134, 2018.
[99] A. H. Hamamoto, L. F. Carvalho, L. D. H. Sampaio, T. Abrão, M. L. Proença Jr, "Network anomaly detection system using genetic algorithm and fuzzy logic, " Expert Systems with Applications, Vol. 92, pp. 390-402, 2018.
[100] R. Sharma, S. Chaurasia, "An Enhanced Approach to Fuzzy C-means Clustering for Anomaly Detection," In: A. Somani, S. Srivastava, A. Mundra, S. Rawat, (Eds.), Proceedings of First International Conference on Smart System, Innovations and Computing, Smart Innovation, Systems and Technologies, Vol. 79. Springer, Singapore, Singapore, pp. 623-636, 2018.
[101] O. Achbarou, M. A. El Kiram, O. Bourkoukou, S. Elbouanani, "A New Distributed Intrusion Detection System Based on Multi-Agent System for Cloud Environment," International Journal of Communication

Networks and Information Security (IJCNIS), Vol. 10, No. 3, pp.526-533, 2018.

Table 5. Number of samples in KDD cup 99 dataset

	Class					
Dataset	**Dos**	**Probe**	**U2R**	**R2L**	**Normal**	**Total**
Corrected KDD 99 (training dataset)	229853	4166	70	1126	60593	**311029**
10% KDD (testing dataset)	391458	4107	52	1126	97277	**494020**

Table 6. Set of selected 12 features from KDD dataset

Attribute Rank	**Attribute number in KDD dataset**	**Attribute name**	**Data Type**
1	3	Service	Symbolic
2	5	Src_bytes	Numeric
3	6	Dst_bytes	Numeric
4	12	Logged_in	Boolean
5	23	Count	Numeric
6	24	Srv_count	Numeric
7	27	Rerror_rate	Numeric
8	28	Srv_rerror_rate	Numeric
9	31	Srv_diff_host_rate	Numeric
10	32	Dst_host_count	Numeric
11	33	Dst_host_srv_count	Numeric
12	35	Dst_host_diff_srv_rate	Numeric

Table 7. Parameters values and performance of best previous ANIDS [32]

		Value
Parameters	Number of attributes	12
	Normalization	Min-Max
	Architecture (I-H-O)	12-10-1
	Method of calculating number of nodes in hidden layer	H= 0.75*Input + Output
	Activation function	Sigmoid
Performance Metrics	Accuracy	98.66%
	Precision	99.62%
	FPR	1.13%
	FNR	1.41%
	TPR (DR)	98.59%
	TNR	98.87%
	F-score	0.99
	AUC	98.73%
	Average time of classification	0.0000890854 s

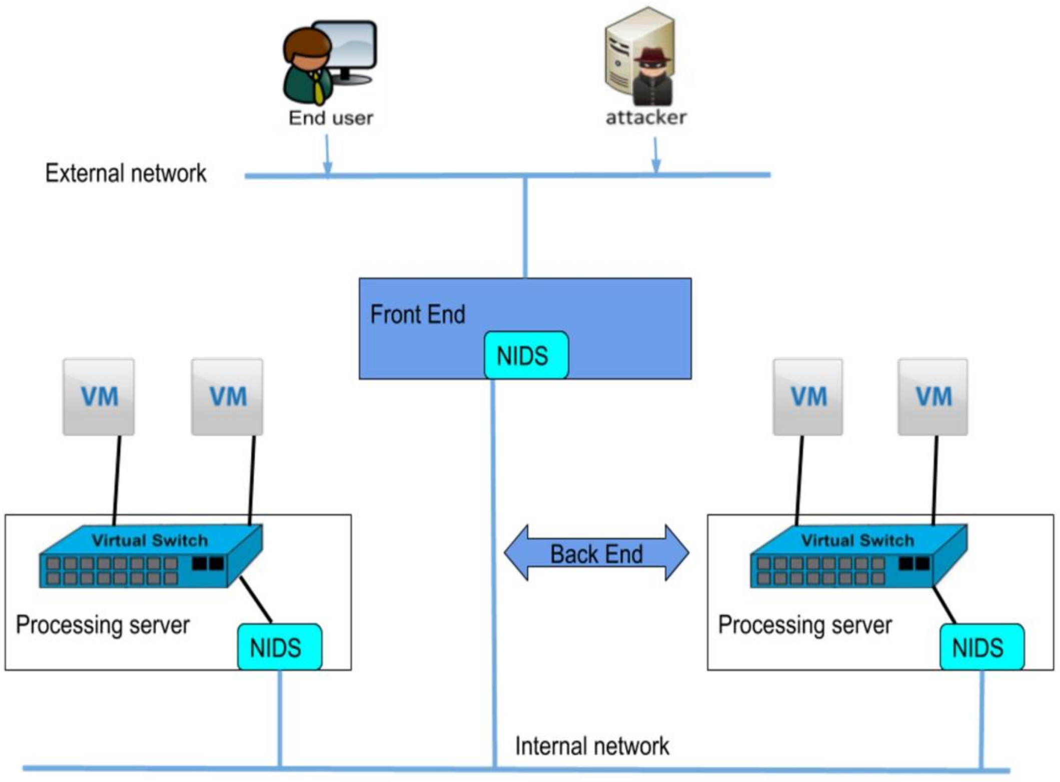


Figure 12. Positions of proposed ANIDS BPNN-IGA in a cloud network

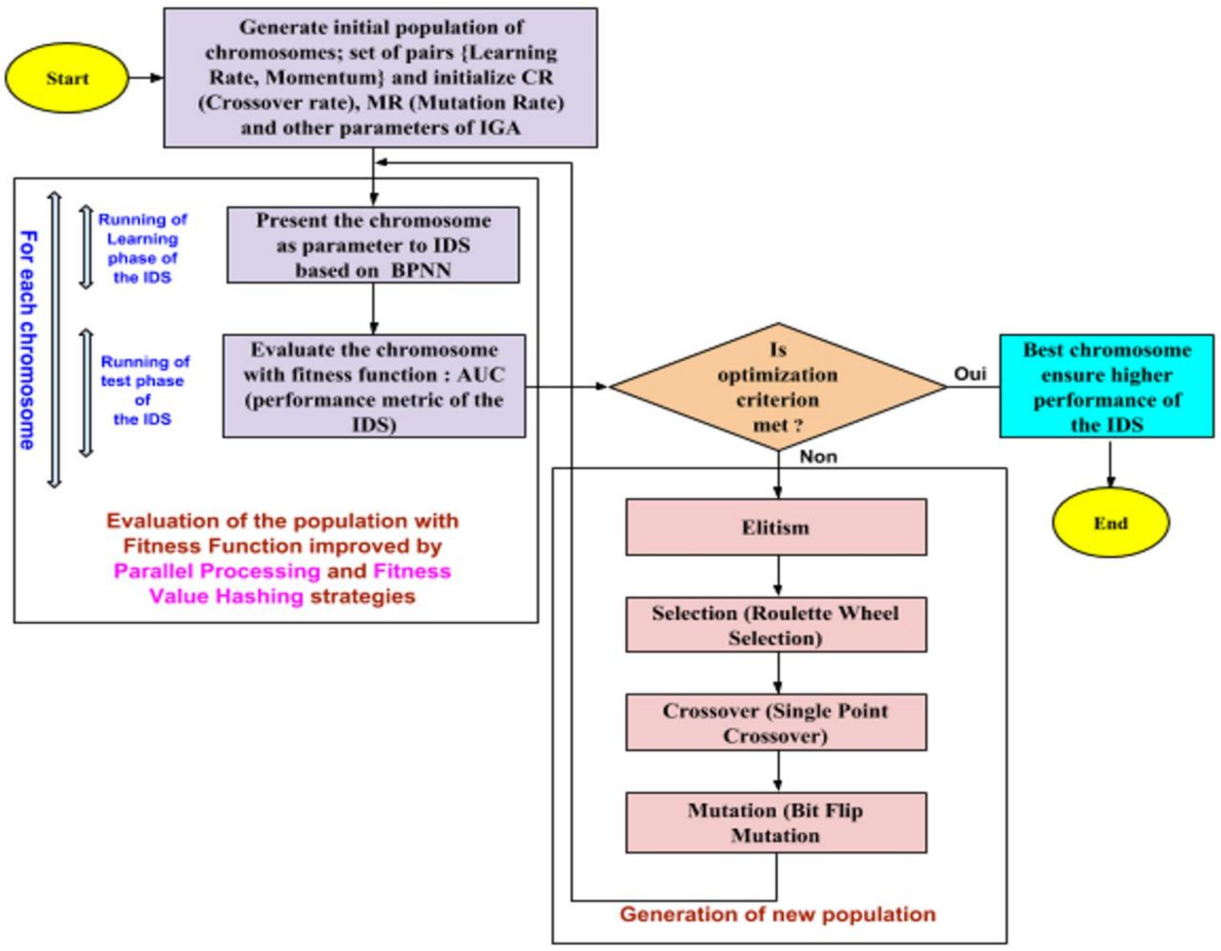

Start
Generate initial population of chromosomes; set of pairs {Learning Rate, Momentum} and initialize CR (Crossover rate), MR (Mutation Rate) and other parameters of IGA
For each chromosome
Running of Learning phase of the IDS
Present the chromosome as parameter to IDS based on BPNN
Running of test phase of the IDS
Evaluate the chromosome with fitness function : AUC (performance metric of the IDS)
Evaluation of the population with Fitness Function improved by Parallel Processing and Fitness Value Hashing strategies
Is optimization criterion met ?
Oui
Non
Best chromosome ensure higher performance of the IDS
End
Elitism
Selection (Roulette Wheel Selection)
Crossover (Single Point Crossover)
Mutation (Bit Flip Mutation
Generation of new population

Figure 13. Workflow of proposed system

Table 8. Comparison of performance of our IDS BPNN-IGA with other works

Research work	Year of publication	Precision	False Positive Rate (FPR)	Accuracy	True Positive Rate (TPR)/ DR	F-Score
A novel Hybrid-Network Intrusion Detection System (H-NIDS) in Cloud Computing [44]	2013	99.60%	1.17%	97.57%	97.14%	0.98
Intrusion Detection and Attack Classification using Back propagation neural network [87]	2014				47.75%	
Applying Hopfield Artificial Network and Simulating Annealing for Cloud Intrusion Detection [88]	2015				<= 93%	
An Efficient Cloud Network Intrusion Detection System [89]	2015			76.54%		
Intrusion Detection System Based on Modified K-means and Multi-level Support Vector Machines [90]	2015		1.45%	95.71%	95.02%	
An Anomaly Detection Model for Network Intrusions Using One-Class SVM and Scaling Strategy [91]	2015	99.03%			91.61%	0.9518
A hybrid method consisting of GA and SVM for intrusion detection system [92]	2016	97.2%	1.7%		97.3%	0.972
An Intrusion Detection System for Detecting Denial-of-Service Attack in Cloud Using Artificial Bee Colony Optimization Algorithm [93]	2016				72.4%	
A New Classification Process for Network Anomaly Detection Based on Negative Selection Mechanism [94]	2016		18%		96%	
Artificial Neural Network Classifier for Intrusion Detection System in Computer Network [79]	2016		5.56%	99.39%	94.05%	
Another fuzzy anomaly detection system based on ant clustering algorithm [95]	2017		10.03%		92.11%	

Two-tier network anomaly detection model- a machine learning approach [96]	2017		4.83%		72.19%	
Fusion of ANN and SVM Classifiers for Network Attack Detection [97]	2017				79.71%	0.75
A Hybrid Methodologies for Intrusion Detection Based Deep Neural Network with Support Vector Machine and Clustering Technique [98]	2018			92.03%	91.35%	
Network Anomaly Detection System using Genetic Algorithm and Fuzzy Logic [99]	2018	95.23%	0.56%	96.53%	76.50%	0.8484
An Enhanced Approach to Fuzzy C-means Clustering for Anomaly Detection [100]	2018	99.4567%		95.2992%	95.764%	
An Enhanced Intrusion Detection System Based on Clustering [19]	2018		25%		90%	
A New Distributed Intrusion Detection System Based on Multi-Agent System for Cloud Environment [101]	2018		12.43%		72.03%	
Our previous IDS BPNN [32]	2018	99.62%	1.13%	98.66%	98.59%	0.99
Our Novel IDS BPNN optimized using Improved Genetic Algorithm (ANIDS BPNN-IGA)		**99.96%**	**0.11%**	**98.82%**	**98.46%**	**0.99**