Computer Systems Science & Engineering
DOI: 10.32604/csse.2023.027216
Article

Tech Science Press

Intrusion Detection Using Federated Learning for Computing

R. S. Aashmi[1,*] and T. Jaya[2]

[1]Department of Computer Science and Engineering, CSI Institute of Technology, Thovalai, Tamilnadu, India
[2]Department of Electronic Communications and Engineering, CSI Institute of Technology, Thovalai, Tamilnadu, India
*Corresponding Author: R. S. Aashmi. Email: aashminishanth@gmail.com
Received: 12 January 2022; Accepted: 02 March 2022

Abstract: The integration of clusters, grids, clouds, edges and other computing platforms result in contemporary technology of jungle computing. This novel technique has the aptitude to tackle high performance computation systems and it manages the usage of all computing platforms at a time. Federated learning is a collaborative machine learning approach without centralized training data. The proposed system effectively detects the intrusion attack without human intervention and subsequently detects anomalous deviations in device communication behavior, potentially caused by malicious adversaries and it can emerge with new and unknown attacks. The main objective is to learn overall behavior of an intruder while performing attacks to the assumed target service. Moreover, the updated system model is send to the centralized server in jungle computing, to detect their pattern. Federated learning greatly helps the machine to study the type of attack from each device and this technique paves a way to complete dominion over all malicious behaviors. In our proposed work, we have implemented an intrusion detection system that has high accuracy, low False Positive Rate (FPR) scalable, and versatile for the jungle computing environment. The execution time taken to complete a round is less than two seconds, with an accuracy rate of 96%.

Keywords: Jungle computing; high performance computation; federated learning; false positive rate; intrusion detection system (IDS)

1 Introduction

Grid Computing is a distributed computing that originated into force with the main purpose of ensuring transparency and accessibility and is considered the best in handling high performance computing applications. As the time flews, the need for better performance in computational power and bandwidth consumption were increased and researchers tried hard to develop the distributed computing paradigms. The evolution of grid computing results in the new trends and technologies like Jungle Computing [1–3], peer-to-peer computing [4], volunteer computing [5], fog computing [6], cloud computing [7] and edge computing [8]. The main intention is to provide a user-friendly, fast and reliable low bandwidth consumption, which can handle high performance computational algorithms and Applications. The resources distributed to the end users maintain efficiency of its functions. The step-up of the internet usage paves the way to set up new kind of devices like iPhone, smart phone, and Internet of Things (IoT)

devices, Light Emitting Diode (LED) television, desktops, laptops, notebooks, tablets and super computers. All these smart devices are utilizing the computing paradigms. These computing prototypes always demand to provide better functionalities like resource utilization, resource scheduling, and accessibility of resources, security and maintenance. The workloads for the computing paradigms are increasing eventually. The intruders find a new type of attacks to interrupt the communication between client and the server. Hence, intrusion detection system becomes one of the most significant techniques that need to address well in these computing paradigms.

Intrusion is the process of detecting the activities of intruders whose action may disrupt the communication between the client and the server. It analyzes the pattern, behavior, content, security policies of the clients and send the request to the server. Internet service provider (ISP) helps to establish the internet services to the customers. Sometimes internet service providers failed to analyze the intention of the customers who are in need of Domain Name Service (DNS). Therefore, some intruders use this advantage and perform intentional attacks to the servers. Present detection methods like signature based intrusion detection, anomaly based intrusion detection has many pros and cons. Hence, there is a need for new learning process, which gives standard security policies and results.

The best solution in jungle computing to tackle intruders is the federated learning technique. Federated learning is one of the machine learning methodology. Low bandwidth consumption devices can have this technique and thereby locally collect the data samples without the interference of computing paradigms**.** The collected data model is transferred to the jungle-computing server and it sends to the global model to all connected devices**.** The traditional machine learning methods collect the data model, upload to the centralized server, and distribute the data samples to all connected devices.

Recently, the application of federated learning technique has been enhanced in the field of defense, medical, big data, telecommunications and Internet of Things (IoT). This technique of learning model has huge contribution in data security, data privacy and handling heterogeneous data. A deep neural network [9] is the backbone of the federated learning algorithm. The ultimate intention of federated learning is that, it understands the local datasets in local data [10], samples and updates standard global data samples to the centralized server. Thus, the global dataset obtains all the latest trends, patterns and behavior of the intruder's data.

The benefits of federated learning for jungle computing include the liability to affect unbalanced, sparse and non-representative data at each smart device. This approach also achieves scalability by leveraging the computing power from devices, which eventually limits the quantity of communication between client-server to make the updated data independent. Privacy is additionally achieved because the updates communicated with the jungle computing main server does not transmit any identifiable information [11].

In this paper, we have implemented a new computing paradigm called jungle computing that can handle low-end spectrum devices to high-end spectrum devices. This jungle computing provides good environment for high performance computational applications and helps in resource independency, middle-ware independency, malleability and fault tolerance over the network. We propose an intrusion detection system, which actively performs in the jungle computing and overcomes all the existing problems in other computing prototypes.

2 Related Work

The authors Hajibaba et al. [12] reviewed the modern distributed computing paradigms like cloud computing, jungle computing and fog computing. They suggested that the jungle computing is heterogeneous and includes clusters grids, clouds, supercomputers, and mobile devices with graphics processing unit (GPU) and Field Programmable Gate Arrays (FPGAS). Once the application starts,

communication problem arise and has solved by characteristics like resource independence, middleware independence, and interoperability. Drost et al. [13] proposed a detailed case study of jungle computing. He analyzed the performance in cluster, grids, clouds, supercomputers, and other platforms and has become more complex, in size, computation and resources. He explained that the computational requirement is very high, and resource availability is very less, also the resources are equal in computing power. Brao et al. [14] proposed the programming is very tough in jungle computing and any changes that occurs in the resources affects the whole program and has to be rewritten. This computing issue is solved using two-stage software, for easy practice. In [15], Zarrin et al. presented a distributed computing, which is very difficult in the recent trends and to solve this jungle computing has emerged a lot. However, the resource is stored in a hierarchal format, which is very difficult to identify for the scientist. To overcome this issue Hybrid Adaptive Resource Discovery (HARD) is proposed and it gives the requirement for high hierarchy, heterogeneity, scalability, and dynamicity, which is self-organized and self-adaptation of resources. The authors Zaghdoudi et al. [16] shown interested with sharing resources in distributed computing. Distributed computing has many advantages like big data, latency-sensitive applications, mobile applications, video-on-demand service, and smart grid. This leads to the emerging of new distributed computing like ad hoc mobile cloud computing, Cloud of Things, Jungle and Fog computing. Liao et al. [17] Says that intrusion means intrude or attacker attacks the security mechanisms of a computer. Intrusion detection is nothing but it automatically starts detecting if there is an intruder in the system. An intrusion prevention system is nothing but preventing the intruder from entering into the system. Elejla et al. [18] says the Internet Control Message Protocol (icmpv6) is very essential due to its functionalities. But it has a high threat of attackers like DDOS (Distributed Denial of Services) and DOS (Denial of service). Packet-based representation and features are the Intrusion Detection System (IDS) which are used commonly but they are not effective, so flow-based representation is used which detects the icmpv6-based DDOS attacks Intrusion Detection. The flow-based approach is based on the flow of network traffic and is distinguished between the normal and the attack flow. Bostani et al. [19], says Internet of things is nothing but connecting to smart devices in the long distance. Communication can be secured by encrypting technique and thereby preventing all attacks is very complex. The most common attack in IoT is routing attacks. Sinkhole and selective forwarding attacks are the common attacks, which are detected by anomaly-based and specification-based intrusion detection modules. Biermann et al. [20], says misuse detection only detects the known attack pattern which in turn gives the high accuracy rate, but does not identify the normal user behavior. If the system knows the details of the attack only then anomaly detection can be used, which has high accuracy and low completeness. IDS operates on a high level of the profile so it was difficult to run on single host. Jeune et al. says [21], internet technology shows immense growth in technology. As users increase, the security became more concern. The organization has to ensure the security of its data. Anomaly based IDS are used to detect the attack in the network traffic by analyzing the packets. NNIV-RS (Neural Network with Indicator Variable using Rough Set for attribute reduction) algorithm was implemented to detect the threat with high accuracy says Sadek et al. [22]. Kumar et al. says [23] Ipv6 is more useful in day to day life. The increase in the host computer in the network needs larger space to make it reliable, but security is the major concerns in this ipv6. To overcome this problem, host-based IDS using active detection technique for ipv6 Neighbor Discovery Protocol (NDP) is used. Aydin et al. [24] says encryption and the firewall were not up to the mark. So Intrusion Detection System (IDS) is used to detect the threat and it identifies the threat when it is occurred. It stores the information about the traffic and secures the information. Intrusion Detection System (IDS) are mainly based on anomaly and misuse-based detection schemes.

3 Technical Background

3.1 Intrusion Detection System

An intrusion detection system (IDS) is one, which greatly helps in monitoring network traffic for fishing movements and gives alarms when detected. It checks for any malicious actions or trespasses in a network. If any, those suspicious actions would be informed to the controller with the help of security information and event management system. This system's main role is to filter alerts from different sources and exclude fake alarms. Therefore, it is much more important for the organization to clearly review and refine the intrusion detection product before feeding it in the system. This includes calibrating the product, teaches IDS to perfectly categorize [25] normal and adversary traffics. This same work is completed by another system called intrusion prevention system, which also alarms when malicious activities are detected.

The main strategy of network intrusion detection system [26] is allotting specific territory in the network, where all the devices within the network territory are scanned for fake traffics. This is effectively done by examining and passing the current traffic to a data set to compare it with the already known attacks, hence the similar ones can be easily rejected. Rejection is confirmed after an alert send to the administrator. Example, network intrusion detection system installed in a subnet of firewall detects, where the firewall is being subjected to break trials.

When host intrusion system [27] is concerned, it is installed only in a particular device, which effectively examines the packets coming in and going out of the device and also detects the fishy moments. It does by comparing the snapshots of current traffic with the existing ones. Talking about protocol based intrusion detection system [28], the entrance of the server is completely monitored, so that it filters the traffic between the user and the server. The web server is protected by frequent examination of the https protocol stream by allowing the later ones. Application protocol based intrusion detection system [29] is mostly supposed to be working within a bunch of servers. The system detects any kind of malicious functions by examining and evaluating the information shared on application which has a specific protocol. As its name suggests, hybrid intrusion detection system [30], works based on two or more types of IDS. Here it uses basic system information or noise with the network data to create an overall understanding of the network system. The combination of hybrid intrusion detection system has lot of advantages over the other types of intrusion detection systems.

3.2 Detection Methods of IDS

Signature based method is used to identify any type of adversary by means of any particular designs. Example, the design of number of bytes on the amount of zeros and ones in the traffic [31]. The style of previously found malicious activities is also taken into account. The design that is found by the intrusion detection system is called as a signature. Therefore, the type of IDS can easily identify the malware attacks that have been already recorded for their patterns, but it is important to be noted that adversaries using new design of attacks are hard to detect.

Anomaly based method is used for finding out adversaries, based on previous records and also by using the present patterns [32]. Machine learning is used to build an activity model, which is considered as genuine, and this model is compared with the incoming ones and those found not matching are marked as attacks. Machine learning based intrusion detection system is more effective than signature-based IDS, as the former is likely to get used to the hardware programs.

3.3 Jungle Computing

The application of high-performance and distributed computing in scientific practice has become more importance among the foremost available platforms like clusters, grids and cloud systems. These infrastructures are now undergoing lot of changes by mixing of core technologies, providing speed

improvements for selected compute kernels. Because of this reason, the distributed and high-performance computing is becoming more heterogeneous and hierarchical, but the complexity in programming is increased. Further, these complexities arise to urgent desire for scalability and issues like data distribution, heterogeneity in software and hardware availability. These issues force scientists into simultaneous use of multiple platforms [33] (e.g., clusters, grids and clouds used concurrently). The usage of computation extends in fields of media analysis [34], remote sensing [35], medical image processing [36], semantic web reasoning [37] and health-care [38].

There are several reasons for using Jungle Computing Systems. The first and foremost reason is, an application may require more computational power than available in its system, in which a user can access. Secondly, different parts of application may have different computational requirements. From a high-level view, all resources during a jungle computer system are equal, all consisting of some amount of processing power, memory and possibly storage. End-users receive a computational resource to run their application. Whether this resource is found during a remote, cloud or located down the hall during a cluster, is of no interest to an end-user, as long as his or her application runs effectively. Despite this similarity of resources, a jungle computer system is very heterogeneous. Resources differ in basic properties like processor architecture, amount of memory and performance. Moreover, for each resource, a special middleware interface must be available, which requires different middleware client software. Once an application has been successfully started during a jungle, another aspect that hinders usage of jungle computing systems is that the lack of connectivity between resources. The architecture of jungle computing is shown in Fig. 1, which clearly depicts the diverse collection of distributed paradigms like clouds, cluster, grids, etc.

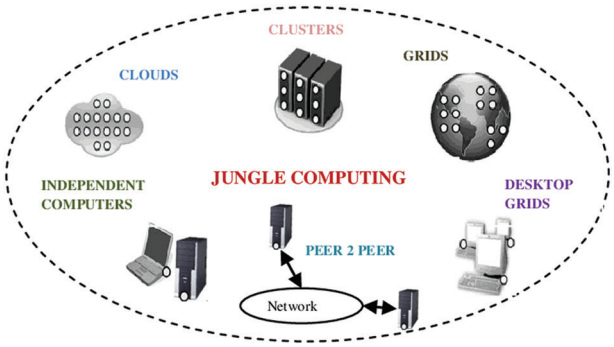

Figure 1: Architecture of jungle computing

High Performance Computation's energy consumption has recently intrigued the scientists because:

1. If a system consumes less energy, it generates less heat. Supported the Arrhenius Law, a component's anticipation decreases by 50% for each 10°C of temperature increase. Therefore, by decreasing, a system's operating temperature, its anticipation is increased [39].
2. Large scale infrastructures require expensive cooling techniques. A method of reducing the amount of cash spent on cooling technologies is to scale back the warmth overall. Thus, big companies like Facebook build their server farms on the sting of the Arctic Circle in Northern Sweden.
3. Authors in [40] have estimated that high performance computation infrastructure will reach 2% of the world's total electricity consumption by the year 2020.

4 Methodology

4.1 Intrusion Detection Using Federated Learning

There are various types of intrusion detection system in the network, however, the existing detection methodologies has increase in number of false positive and false negative rates. It also need human intervention either to update signatures or to learn the sub sequential correlation between transactions [41]. The existing machine learning methods has fixed data models which complicates further updating of models. This paves the way for new developed attacks into the network, even before realizing it.

Federated Learning can be used in many fields, because it ensures lower latency, less power consumption and higher privacy. Federated learning is also known as collaborative learning, which measures the set of experiences on-gadget to propose enhancements to the following emphasis of new idea model. To make federated learning feasible, we need to overthrow numerous algorithmic and specialized difficulties. In a conventional AI framework, an advancement calculation like Stochastic Gradient Descent (SGD) runs on a large dataset divided homogeneously across servers in the cloud. Such exceptionally iterative calculations require low-latency, high-throughput associations with the preparation of data.

Federated learning completely set its place over all the present methodologies. Initially, we extract the dataset from Canadian Institute for Cyber security (CIC) project funded by Canadian Internet registration authority (CIRA)-2020 [42]. Then we update the model by learning from each connected devices with the computing paradigms. The implemented Federated Learning has to undergone these steps.

- Federation Construction
- Decentralized Training
- Model Accumulation
- Model Aggregation

During first step, Pre-trained model is implemented to the connected devices. These models are derived from CIRA-CIC-DoHBrw-2020. An odd subset of stakeholders are preferred to get the global model synchronously from the server. As the decentralized training is processed, each selected devices with jungle computing in the federated construction subjected to enumerate an updated model by analyzing the behavior, content and domain name system (DNS) of the requests received to that particular device. This data is called as local data. Then model accumulation is performed as the third step. In this step, the gathered local data is analyzed and derive out an updated data or model from it.

These updates are sent back to the server. However, the other data remains unchanged in the server. Finally, model aggregation is performed in the federated learning mechanism. The server aggregates the updated model or data from the model accumulation. By applying the federated average (FedAvg), the model weights transpire averaged to develop an enhanced version of global model. This global model is distributed to all its connected devices in the jungle computing. These operations are intended to enforce the same procedures repeatedly.

Algorithm 1: Federated Averaging. C is the global batch size; B is the local Batch size; the K clients are indexed by k; E is the number of local epochs; and η *is the learning rate*

procedure SERVERUPDATE:

initialize ω_0

for each round $t = 1, 2, \ldots, do$

$$m \leftarrow max(C.\ K,\ 1)$$

$$S_t \leftarrow (subset\ m\ of\ clients)$$

For each client $k \in S_t$ ***in parallel do***

$$\omega_{t+1}^k \leftarrow ClientUpdate(k\ \omega_t)$$

end for

$$\omega_{t+1}^k \leftarrow \Sigma_{k=1}^k \frac{n_k}{n} \omega_{t+1}^k$$

end for

end procedure

procedure CLIENTUPDATE (k, ω) // Run on client k

$$B \leftarrow (split\ data\ k\ into\ batches\ of\ size\ B)$$

for each local epoch t from 1 to E **do**

for batch $b \in B$ ***do***

$$\omega \leftarrow \omega - \eta\ \Delta F_k(\omega)$$

end for

end for

return ω to server

end procedure

Federated learning preferred to get the global model synchronously as the main memory for storage of files, where there is a need for local data and global data. Local data is a collection of files that contains the traffic record on daily basis by calculating the number of times it has been used. Global data is a traffic and individual record of data memory where the most used files are eventually updated by averaging the collected local data. Federated construction is subjected to collect the files, which are used on daily basis depending upon the file accessed time, known as local data. Decentralized training analyzes the local data. Model accumulation accumulates the local data and performs the operation of categorization to derive out an updated model from it. These updates are sent back to the main storage in the memory and then it is subjected to the model aggregation, where the model weights are averaged to an enhanced version of global model. The replacement of files is performed according to the model weight. Model weight is the calculation of the data that describes the number of times a file accessed by the user and is shown in Fig. 2.

4.2 Implementing Federated Learning in Jungle Computing

The implementation of federated learning to detect intrusion with jungle computing is quite simple and elegant. In order to run multiple computing paradigms at the same time, it requires an efficient computing called jungle computing. All the devices of IoT- slot1a, IoT- slot1b, IoT- slot1c devices are connected with the gateway through Bluetooth. In the gateway, the intrusion detection system using federated learning takes place. Each IoT devices itself has the ability to gather local data models and sent an

updated model to the server in the jungle computing through the gateway. The updated model known as global model is again sent to all the devices connected with the jungle computing. Model repository collects the overall dataset needs to proceed imminent control, as shown in Fig. 3. The following procedure takes place:

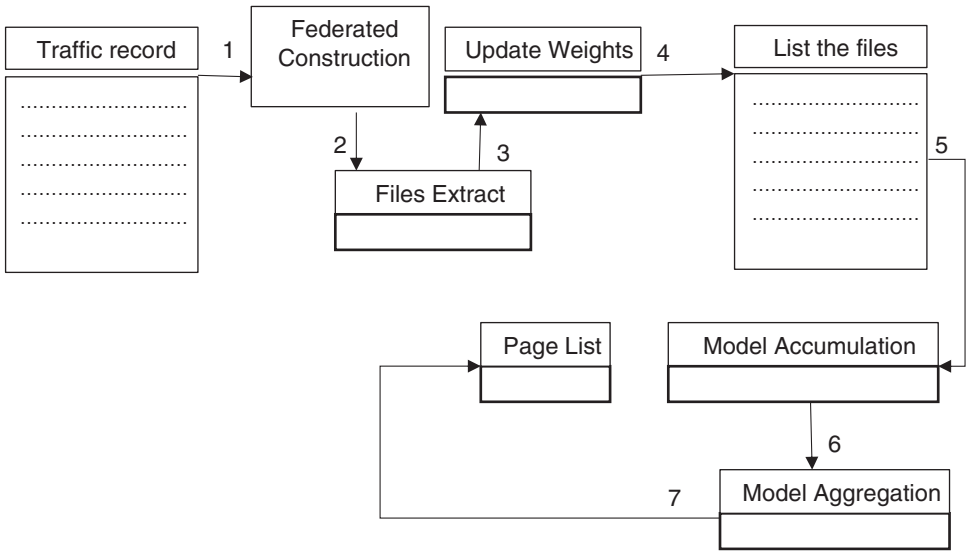

Figure 2: Design flow of federated learning

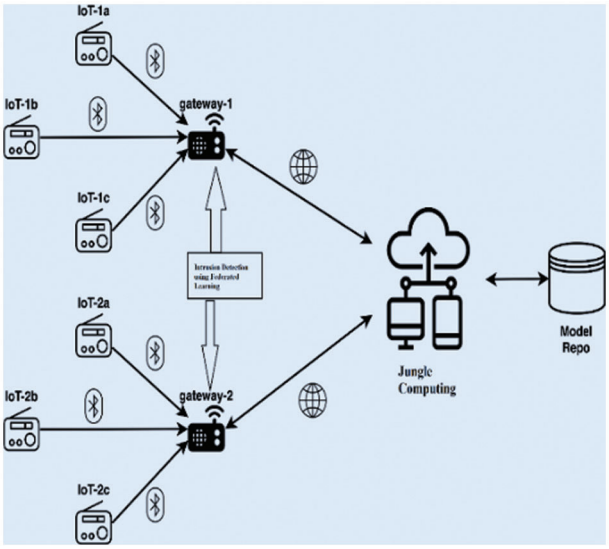

Figure 3: Architecture of proposed system

1. Capture data.
2. Construct feature matrix and vectors.
3. Train/Test Split.
4. Setup environment.
5. Prepare federated data.

6. Train model in federated way.
7. Save, Load, Predict.

5 Experimental Results

In our experimental tests, we demonstrate the viability of the FedAvg calculation for the intrusion detection issue. Combining local models by averaging their weights on the central server functions works well in any straightforward neural network model. For this situation, the input layer has various neurons equivalent to the quantity of characteristics in the dataset and the yield layer has various neuron equivalent with the absolute number of classes. Tests are made in the design that shows the distinction among highlights and will be clarified in particular subsections. The attributes used for feature matrix in the dataset are 'duration', 'src_bytes', 'dst_bytes', 'wrong_fragment', 'urgent', 'hot','num_failed_logins', 'num_compromised', 'root_shell', 'su_attempted', 'num_root', 'num_file_creations', 'num_shells', 'num_access_files', 'num_outbound_cmds', 'count', 'srv_count', 'serror_rate', 'srv_serror_rate', 'rerror_rate', 'srv_rerror_rate', 'same_srv_rate', 'diff_srv_rate', 'srv_diff_host_rate', 'dst_host_count', 'dst_host_srv_count', 'dst_host_same_srv_rate', 'dst_host_diff_srv_rate', 'dst_host_same_src_port_rate', 'dst_host_srv_diff_host_rate', 'dst_host_serror_rate', 'dst_host_srv_serror_rate', 'dst_host_rerror_rate', and 'dst_host_srv_rerror_rate. The experiment results of intrusion detection using federated learning initially needs learning_rate = 1, n_epochs = 500, n_clients = 5, client_batch_size = 5, n_rounds = 10. And each round requires an average client weight with their own batch size to compute the model weight. As well as to calculate local data and global data derive the updated global data. This examination shows that expanding the boundaries bring about better precision, in return of perusing more bytes in each round and altogether. In the greater part of the test cases, we have accomplished a 90–96% precision in differentiating malicious attacks if test set is prepared every so often areas.

The untrained accuracy *vs.* trained accuracy is calculated by using federated learning and the accuracy using updated weights are estimated. Experiment results of intrusion detection using federated learning is shown in Fig. 4.

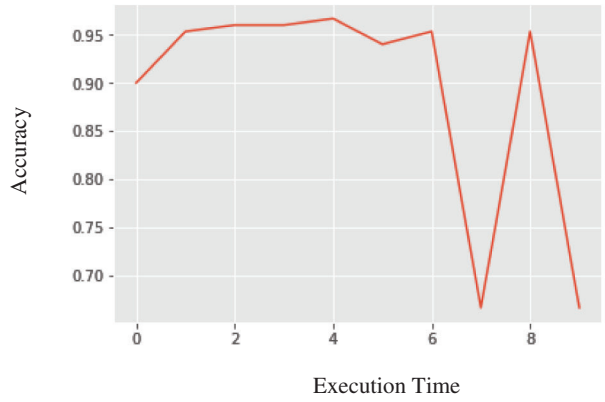

Figure 4: Experiment results of intrusion detection using federated learning

- round 0 execution time: 3. 1388421058654785 accuracy: 0. 8999999761581421
- round 1 execution time: 2. 8116440773010254 accuracy: 0. 95333331823349
- round 2 execution time: 2. 782147169113159 accuracy: 0. 9599999785423279
- round 3 execution time: 2. 7059502601623535 accuracy: 0. 9599999785423279

- round 4 execution time: 2. 756247043609619 accuracy: 0. 9666666388511658
- round 5 execution time: 2. 7820329666137695 accuracy: 0. 9399999976158142
- round 6 execution time: 2. 8027403354644775 accuracy: 0. 95333331823349
- round 7 execution time: 2. 7404510974884033 accuracy: 0. 6666666865348816
- round 8 execution time: 2. 8029439449310303 accuracy: 0. 95333331823349
- round 9 execution time: 2. 7011499404907227 accuracy: 0. 6666666865348816

6 Performance Evaluation

The performance evaluation for both Federated learning and jungle computing needs to be elaborated to understand the overall mechanism of the model. The accuracy of the model increments and the false positives rate diminishes for all attacks during the training of this federated algorithm. Therefore, we can presume that we made a measurably huge and positive effect of the detection accuracy for detecting attacks. A harmony point being reached in the federated algorithm where every portable devices begins with the federated model, figures the update, also gets back to the server by creating a federated model. Communication cost of the federated algorithm is estimated by the quantity of bytes communicated to and from the handy devices and are shown in Fig. 5. The performance of our proposed federated algorithm to detect all kinds of intrusion attacks is estimated by correlation with other non-Federated learning algorithms like Nearest Neighbor (KNN), Decision Trees (DT), Stochastic Gradient Descent (SGD), Support Vector Machine (SVM) and Neural Network (NN). The results shown in Fig. 6 which displays the KNN, DT, and SGD classifiers are substantially more time performing than SVM, neural networks, and the federated algorithm. The training time is equivalent to the SGD classifier and 26. 8 occasions quicker than the non-federated NN. Fig. 7, explains the efficiency of attack detection by non-federated and federated algorithm.

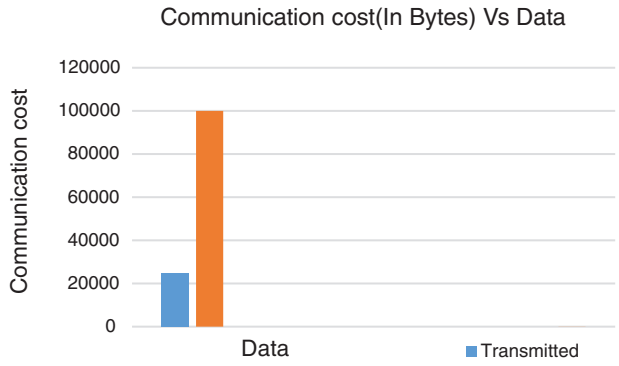

Figure 5: The communication cost of transmitted data size and trained data size

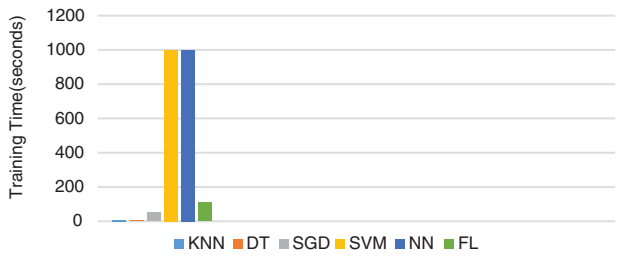

Figure 6: Comparison of the training time taken by the Non-federated algorithm *vs.* Federated algorithm

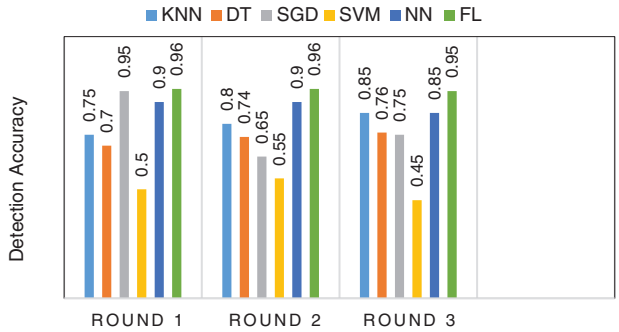

Figure 7: Efficiency of attack detection by non-federated and federated algorithm

The performance of a jungle computing system with the utilization of a load balancer, controls the amount of turned on virtual machines and low-cost energy machines, in order to attenuate expenses such as: the value of employing a cloud system and therefore the cost of energy consumption. We assume that the local cluster is usually powered, so we will not minimize energy but we can utilize its computation power to attenuate the necessity resources when the system is not loaded. Here we have utilized jungle computing running high-performance applications by using the java parallel processing framework. The graphs shown in Figs. 8, 9 explain the machine workload *vs.* the speed of the different computing paradigms. Normalized speed also evaluated during the performance. The communication overhead also evaluated for virtual machines like java/ibis/TCP, java/ibis/MX, C++/MPI/MX suitable for multiple platforms clusters, grids, and clouds programing.

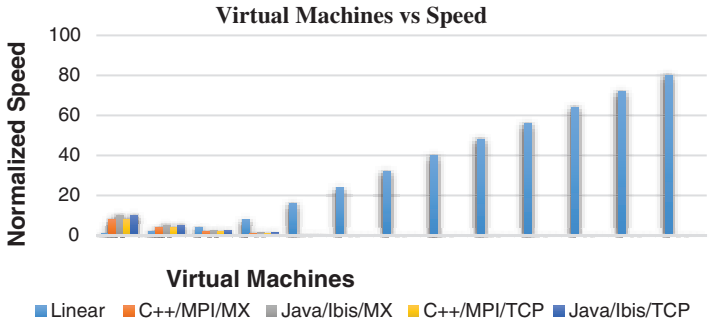

Figure 8: Performance and speed characteristics of all application versions

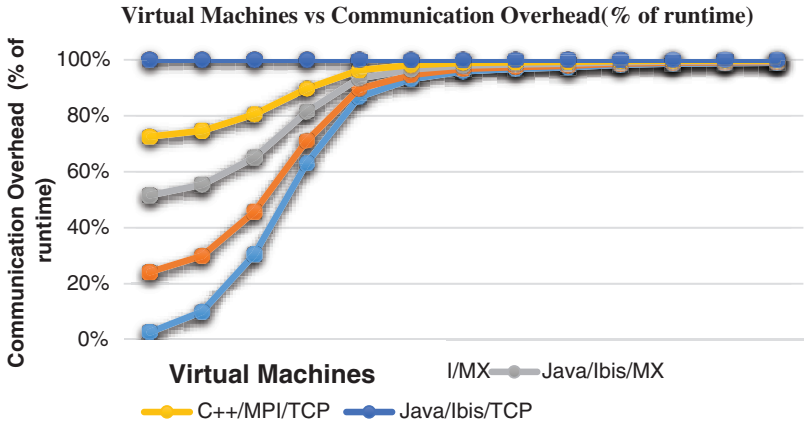

Figure 9: Communication overhead of all application versions

7 Conclusion

In this paper, we have proposed an efficient Intrusion detection system using Federated learning (FL) in Jungle Computing (JC). Our approach includes the involvement of Jungle computing which provides scalability, accessibility, Fault tolerance and handles multiple computing paradigms at a time. Jungle computing helps to run the high-performance applications by utilizing the Java Parallel Processing Framework. Moreover, the meta-tags will be attached to jobs, to define which jobs are often executed. By simply changing the meta-tags of jobs, the Jungle's computer system architecture are often altered and therefore the application can run efficiently in every possible combination of resources, desktop PCs, laptops, clusters, grids, cloud, low cost devices, smartphones: a real computing jungle. This proposed approach incorporates an intrusion detection system using federated-learning mechanism toward jungle computing by applying java parallel processing framework. Privacy-enhanced federated learning frameworks are more robust to anomaly detection models. The reason is that the federated learning framework is susceptible to malicious attacks by malicious participants and a more robust model applied to a wider range of application scenarios in jungle computing by exercising java parallel processing framework. This implementation prevents the intrusion in the network by utilizing the Federated learning in jungle computing that has greatly enhanced the efficiency. The 96% of attacks in the network discovers at a fraction of milliseconds on a regular basis with no levitation of any counterfeit alarms. The false positive rate decreases enormously while compared to other non-federated algorithms and hence this method is superior in its state-of-art works.

Funding Statement: The authors received no specific funding for this study. The authors will give the entire processing fee needed for this journal.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] D. Tychalas and H. Karatza, "High performance system based on cloud and beyond: Jungle computing," *Elsevier: Journal of Computational Science*, vol. 22, pp. 131–147, 2017.

[2] I. Foster, C. Kesselman and S. J. Tuecke, "The anatomy of the grid: Enabling scalable virtual organizations," *International Journal of High-Performance Computing Applications*, vol. 15, no. 3, pp. 200–222, 2001.

[3] M. Cafaro and G. Aloisio, "Grids, clouds and virtualization, computer communications and networks," in *Springer-Verlag*, 2011th ed., London: Springer, 2011.

[4] J. Khan and A. Wierzbicki, "Guest editor's introduction; foundation of peer-to-peer computing," *Elsevier Computer Communications*, vol. 31, no. 2, pp. 187–189, 2008.

[5] Z. Zhao, F. Yang and Y. Xu, "PPVC: A P2P volunteer computing system," in *Proc. of the 2nd IEEE Int. Conf. on Computer Science and Information Technology*, Kuala Lumpur, Malaysia, pp. 51–55, 2009.

[6] M. E. Idrissi, O. Elbeqqali and J. RIFfi, "A review on relationship between Iot–cloud computing–fog computing (Applications and challenges)," in *Proc. of the Third Int. Conf. on Intelligent Computing in Data Sciences (ICDS)*, Marrakech, Morocco, pp. 1–7, 2019.

[7] L. Alhenaki, A. Alwatban, B. Alamri and N. Alarifi, "A survey on the security of cloud computing," in *Proc. of the 2nd Int. Conf. on Computer Applications & Information Security (ICCAIS)*, Riyadh, Kingdom of Saudi Arabia, pp. 1–7, 2019.

[8] C. Song, M. Zhang, Y. Zhan, D. Wang, L. Guan *et al.,* "Hierarchical edge cloud enabling network slicing for 5G optical fronthaul," *Journal of Optical Communications and Networking*, vol. 11, no. 4, pp. B60–B70, 2019.

[9] H. B. McMahan, E. Moore, D. Ramage, S. Hampson and B. A. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. of the 20th Int. Conf. on Artificial Intelligence and Statistics (AISTATS)*, Fort Lauderdale, Florida, USA, vol. 54, 2017.

[10] Y. Liu, X. Yuan, Z. Xiong, J. Kang, X. Wang *et al.,* "Federated learning for 6G communications: Challenges, methods, and future directions," *China Communications*, vol. 17, no. 9, pp. 105–118, 2020.

[11] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi and M. Ghogho, "Deep learning approach for network intrusion detection in software defined networking," in *Proc. of the Int. Conf. on Wireless Networks and Mobile Communications (WINCOM)*, Fez, Morocco, pp. 258–263, 2016.

[12] M. Hajibaba and S. Gorgin, "A review on modern distributed computing paradigms: Cloud computing, jungle computing and fog computing," *Journal of Computing and Information Technology*, vol. 22, no. 2, pp. 69–84, 2014.

[13] N. Drost, J. Maassen, M. V. Meersbergen, H. E. Bal, F. I. Pelupessy *et al.,* "High-performance distributed multi-model/Multi-kernel simulations: A case-study in jungle computing," in *Proc. of the (Institute of Electrical and Electronics Engineers) IEEE 26th Int. Parallel and Distributed Processing Symp. Workshops & PhD Forum*, Shanghai, China, pp. 150–162, 2012.

[14] B. Brao and K. Swathi, "Fast KNN classifiers for network intrusion detection *s*ystem," *Indian Journal of Science and Technology*, vol. 10, no. 14, pp. 1–10, 2017.

[15] J. Zarrin, R. L. Aguiar and J. P. Barraca, "HARD: Hybrid adaptive resource discovery for jungle computing," *Elsevier Journal of Network and Computer Applications*, vol. 90, pp. 42–73, 2017.

[16] B. Zaghdoudi, H. K. Ayed and I. Gnichi, "A protocol for setting up ad hoc mobile clouds over spontaneous MANETs: A proof of concept," in *2016 Cloudification of the Internet of Things (CIoT)*, Paris France, pp. 1–6, 2016.

[17] H. J. Liao, C. H. R. Lin, Y. C. Lin and K. Y. Tung, "Intrusion detection system: A comprehensive review," *Elsevier Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[18] O. E. Elejla, M. Anbar, B. Belaton and B. O. Alijla, "Flow-based IDS for ICMPv6-based DDoS attacks *d*etection," *Arabian Journal for Science and Engineering*, vol. 43, no. 12, pp. 7757–7775, 2018.

[19] H. Bostani and M. Sheikhan, "Hybrid of anomaly-based and specification-based IDS for internet of things using unsupervised OPF based on map reduce approach," *Elsevier Computer Communications*, vol. 98, pp. 52–71, 2017.

[20] E. Biermann, E. Cloete and L. Venter, "A comparison of intrusion detection systems," *Elsevier Computers & Security*, vol. 20, no. 8, pp. 676–683, 2001.

[21] L. L. Jeune, T. Goedemé and N. Mentens, "Machine learning for misuse-based network intrusion detection: Overview, unified evaluation and feature choice comparison framework," *(Institute of Electrical and Electronics Engineers) IEEE Access*, vol. 9, pp. 63995–64015, 2021.

[22] R. A. Sadek, M. S. Soliman and H. S. Elsayed, "Effective anomaly intrusion detection system based on neural network with indicator variable and rough set reduction," *International Journal of Computer Science Issues*, vol. 10, no. 6, pp. 227–233, 2013.

[23] N. Kumar, G. Bansal, S. Biswas and S. Nandi, "Host based IDS for NDP related attacks: NS and NA spoofing," in *2013Annual (Institute of Electrical and Electronics Engineers) IEEE India Conf. (INDICON)*, Mumbai, India, pp. 1–6, 2013.

[24] M. A. Aydın, A. H. Zaim and K. G. Ceylan, "A hybrid intrusion detection system design for computer network security," *Elsevier Computers & Electrical Engineering*, vol. 35, no. 3, pp. 517–526, 2009.

[25] D. A. Effendy, K. Kusrini and S. Sudarmawan, "Classification of intrusion detection system (IDS) based on computer network," in *Proc. of the 2nd Int. Conferences on Information Technology, Information Systems and Electrical Engineering (ICITISEE)*, Yogyakarta, pp. 90–94, 2017.

[26] A. Halimaa and K. Sundarakantham, "Machine learning based intrusion detection system," in *Proc. of the 3rd Int. Conf. on Trends in Electronics and Informatics (ICOEI)*, Tirunelveli, India, pp. 916–920, 2019.

[27] U. Bashir and M. Chachoo, "Intrusion detection and prevention system: Challenges & opportunities," in *Proc. of the Int. Conf. on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, pp. 806–809, 2014.

[28] K. M. Yu, M. F. Wu and W. T. Wong, "Protocol-based classification for intrusion detection," in *Proc. of the 7th WSEAS Int. Conf. on Applied Computer & Applied Computational Science (ACACOS '08)*, Hangzhou, China, April 6–8, pp. 749–754, 2008.

[29] M. Labonne, A. Olivereau, B. Polvé and D. Zeghlache, "Unsupervised protocol-based intrusion detection for real-world networks," in *Proc. of the ICNC 2020: Int. Conf. on Computing, Networking and Communications*, Big Island, United States, pp. 299–303, 2020.

[30] C. Cortes and D. Pregibon, "Signature-based methods for data streams," *Data Mining and Knowledge Discovery*, vol. 5, pp. 167–182, 2001.

[31] R. Samrin and D. Vasumathi, "Review on anomaly based network intrusion detection system," in *Proc. of the Int. Conf. on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, Mysuru, India, pp. 141–147, 2017.

[32] W. Zhu, C. Luo, J. Wang and S. Li, "Multimedia cloud computing," *(Institute of Electrical and Electronics Engineers) IEEE Signal Process*, vol. 28, no. 3, pp. 59–69, 2011.

[33] M. Shi, F. Xie, Y. Zi and J. Yin, "Cloud detection of remote sensing images by deep learning," in *Proc. of the (Institute of Electrical and Electronics Engineers) IEEE Int. Geoscience and Remote Sensing Symp. (IGARSS)*, Beijing, China, pp. 701–704, 2016.

[34] C. Yang, L. Chen, W. Chou and K. Wang, "Implementation of a medical image file accessing system on cloud computing," in *Proc. of the 13th (Institute of Electrical and Electronics Engineers) IEEE Int. Conf. on Computational Science and Engineering*, Hong Kong, China, pp. 321–326, 2010.

[35] P. Mika and G. Tummarello, "Web semantics in the clouds," *IEEE Intelligent Systems*, vol. 23, no. 5, pp. 82–87, 2008.

[36] D. Tychalas and H. Karatza,""A cloud system for health care," in *Proc. of the 19th Panhellenic Conf. on Informatics*, Athens Greece, pp. 169–170, 2015.

[37] B. Kahanwal and T. P. Singh, "The distributed computing paradigms: P2P, grid, cluster, cloud, and jungle," *International Journal of Latest Research in Science and Technology*, vol. 1, no. 2, pp. 183–187, 2012.

[38] G. Terzopoulos and H. Karatza, "Performance evaluation and energy consumption of a real-time heterogeneous grid system using DVS and DPM," *Elsevier Simulation Modelling Practice and Theory*, vol. 36, pp. 33–43, 2013.

[39] R. Peng and X. Wang, "ICT solutions calculation model for CO_2 emission reduction and prediction on its emission reduction potential," in *Proc. of the 2009 Int. Conf. on Management and Service Science*, Wuhan, Beijing, China, pp. 1–5, 2009.

[40] N. Drost, J. Maassen, M. A. J. V. Meersbergen, H. E. Bal, F. I. Pelupessy *et al.,* "High-performance distributed multi-model/Multi-kernel simulations: A case-study in jungle computing," in *Proc. of the (Institute of Electrical and Electronics Engineers) IEEE 26th Int. Parallel and Distributed Processing Symp. Workshops & PhD Forum*, Shanghai, China, pp. 150–162, 2012.

[41] I. Sharafaldin, A. H. Lashkari and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization," in *Proc. of the 4th Int. Conf. on Information Systems Security and Privacy (ICISSP 2018)*, Portugal, pp. 108–116, 2018.

[42] M. MontazeriShatoori, L. Davidson, G. Kaur and A. H. Lashkari, "Detection of DoH tunnels using time-series classification of encrypted traffic," in *Proc. of the IEEE Int. Conf. on Dependable, Autonomic and Secure Computing, Int. Conf. on Pervasive Intelligence and Computing, Int. Conf. on Cloud and Big Data Computing, Int. Conf. on Cyber Science and Technology Congress (DASC/PiCom/CBDCom/CyberSciTech)*, Calgary, AB, Canada, pp. 63–70, 2020.