International Journal for Multidisciplinary Research (IJFMR)
E-ISSN: 2582-2160 ● Website: www.ijfmr.com ● Email: editor@ijfmr.com

Study of Simulation Tools for Various Computing Paradigms: Cloud-Fog-Edge

Anu[1], Jyoti Yadav[2]

[1]*Research Scholar, Deenbandhu Chhotu Ram University of Science and Technology, Murthal, Sonipat, Haryana, India*
[2]Assistant Professor, Dr. B. R. Ambedkar Government College, Dabwali

Abstract
The transition from cloud to edge is an intricate and extremely dense scenario. The cloud has an abundance of available resources., whereas, at the edge, they are extremely limited. Because of this complexity, it is necessary to employ reliable simulators to evaluate and assess the performance of innovative mechanisms, as these mechanisms cannot be evaluated in real-world environments. As a result, simulation is a viable option for conducting exploratory tests before making an investment in high-priced and technically advanced real-world testbeds. This research compares and contrasts several different modeling methods, focusing on the betterment of the future. Researchers and developers can benefit from this study's analysis and review of Cloud-Fog-Edge computing simulators by testing out different scenarios and dealing with real-world problems before committing to a certain simulation tool for their research.

Keywords: Simulator**,** Cloud, Fog, Edge, IoT

I. INTRODUCTION

Recent years have seen a shift toward cloud-based data storage and computing, enabled by paradigms that facilitate the management of the massive amount of data that users generate regularly. Furthermore, new difficulties have arisen because of the necessity to construct novel systems capable of handling the data generated by smart gadgets as their prices have reduced. The Internet of Things (IoT) concept was developed to read, store, and handle the huge volumes of data that all these gadgets utilize or generate regularly[1]. Since the amount of data being generated quickly outpaced the capacity of the network, no effective processing paradigms had been developed until recently. As a result, new computing models, including the Cloud, the Edge, and the Fog, have evolved [2]. As a first step in alleviating these issues, cloud computing was adopted to distribute data among the various computers that make up the infrastructure to ease overcrowding and quicken the processing of data. In the wake of cloud computing's global adoption as the dominant model for data management, technological advancements have facilitated the widespread adoption of smart devices, dramatically increasing the need for bandwidth and computing power across networks. This prompted the development of cutting-edge computing paradigms like fog and edge computing, which pushed data processing to the network's periphery in order to improve response times, decrease latency, and expand access to data.

The edge computing paradigm delivers data processing and storage to the network's edge [3]. This paradigm includes mobile cloud computing and fog computing elements. The fog paradigm is a multi-

layered concept that aims to provide appropriate, on-demand access to a large set of dynamic computational resources. Fog nodes, positioned between the cloud and the edge of the IoT network, allow for the deployment of dispersed and latency-aware applications and services while maintaining the benefits of the cloud [4]. In order to effectively accomplish the characteristics of these computing paradigms, it is necessary to conduct increasing amounts of research into cloud-fog-edge environments. To determine the effectiveness of these new paradigms, they must be evaluated. Therefore, the majority of the researchers' efforts are concentrated on techniques such as testbeds and simulators.

II. MOTIVATION

A simulation can demonstrate the evolution of a framework. Rate sufficiency and the execution of complex frameworks are two areas where simulation is frequently utilized. Simulation devices in computing paradigms are essential for execution verification, productivity, and application reliability. Several simulation tools are being developed and evaluated for potential use in computing environments. Every computing paradigm has its own device requirements, which a specific simulator cannot fulfill. This work offers a conceptual review of several simulators of the cloud, fog, and edge paradigm so that researchers can select the best suitable simulator for their problem objective.

III. CONTRIBUTION

This paper emphasizes on efficient simulation tools for different computing paradigms. In this study, we comprehensively analyze existing simulation tools for edge, fog, and cloud computing. The main contributions of this study are summarized below:

1. Identified the primary research issues in the field and the methods whereby simulation may assist in their resolution.
2. Presented an in-depth analysis of cloud-fog-edge computing simulators, highlighting their distinctive features.
3. The pros and cons of available simulation tools were highlighted.

IV. ORGANIZATION

The subsequent sections of this paper are organized as follows: section V presents a background study of Cloud, Fog, and Edge Computing Integration. Various simulators are presented in section VI, and their analysis is offered in section VII, followed by the conclusion in section VIII.

V. CLOUD-FOG-EDGE COMPUTING INTEGRATION

Different architectures are widely utilized in research because there is no standard design for cloud-fog-edge computing [5] [6]. The cloud-fog-edge computing infrastructure is depicted conceptually in Figure 1. The most common implementation includes the following three layers:

1) Cloud Layer

A large number of powerful computers and storage devices make up the cloud computing layer. It provides a number of application services for highly automated and complex systems, including smart homes and futuristic workplaces. It has the computational power to process and analyze massive amounts of data and the storage capacity to keep that data safe for an extraordinarily long time. Fog computing is an alternative approach to the conventional cloud computing model in which all data

processing and storage is done in the cloud. Cloud-based resource management software regulates the entire system and provides quality service to fog computing programs.

In this setup, all client devices are connected to the fog nodes via wired or wireless connections. Connectivity between fog nodes can be established using wired or wireless channels. The IP backbone connects each fog node to the cloud.

2) Fog Layer

Fog is positioned between the terminal layer and the cloud layer to provide separation between the two. A thick layer of fog exists between the client devices and the cloud. In order to acquire services, the end devices can immediately connect to fog nodes. This layer is useful for low-latency applications and real-time analysis. The fog nodes collaborate with the cloud data centers to deliver a more resilient storage and computing infrastructure. The fog nodes and cloud data centers are connected by the IP core network.

3) Edge Layer

Sensors, smart cards, smartphones, and smart vehicles are all examples of IoT or intelligent devices that fall into this layer, which is the one most directly accessible to end users. In most cases, these gadgets are spread out in different locations. They are responsible for recording and relaying feature information about physical objects and events to a higher layer for further processing and storage. Applications that can be installed on the end devices via download and installation make this layer. The next-layer network is utilized by terminal layer devices to provide end-to-end device-to-cloud connectivity.

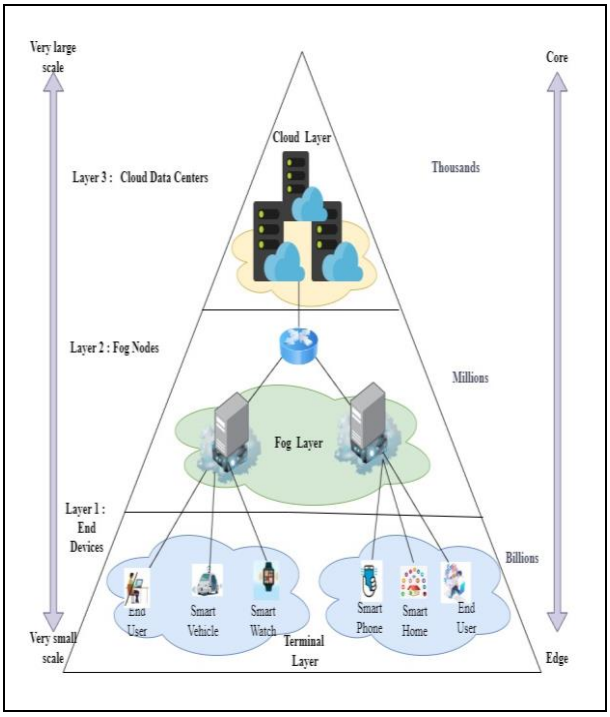

Fig 1. Cloud-FOG-Edge Integration [6]

VI. SIMULATION TOOLS

Cloud, fog, and edge computing evolution have increased the number and complexity of IoT devices. To simulate these environments and assess how well they function in terms of the amount of resource utilization, the amount of time spent, and the amount of energy consumed, a simulation platform is required. Multiple simulators are available, with each offering a unique set of capabilities. It is possible

that a simulator that works well for one kind of research job will not work well for another. If you consider using the same simulator for all your research despite the fact that each one has different requirements, you could end up with unneeded delays. Consequently, the selection of a suitable simulator is very challenging. This paper aims to compare and contrast some of the latest and most widely used simulators available. The following are some of the existing simulators utilized by the researchers to model various computer systems:

A. *CloudSim*

CloudSim[7] is a popular approach for simulating and evaluating cloud situations. It is one of the most well-known and widely used simulators by researchers. CloudSim is used by academic and business developers to try out new application services in a risk-free, straightforward setting. Users of CloudSim can program their own cloudlets to perform various activities, which are then carried out by virtual computers using cloud services. The researchers at CLOUDS Lab introduced this new Java-based system. CloudSim has seen multiple releases, the most recent being version 6.0.0 beta.

B. *CloudAnalyst*

The CloudAnalyst[8] tool was built upon the foundation of CloudSim. It's a simple program that simulates and visualizes massive cloud-based software deployments. It allows for the specification of application workloads, such as location, number of data centers, total users employed, and resource distribution across those locations. The purpose of CloudAnalyst is to decouple the two activities of programming and simulation. This will allow researchers to focus on the intricacy of the simulation without having to spend excessive time on the simulation's coding nuances.

C. *iFogSim*

Simulating fog and cloud resource management and application scheduling with iFogSim are possible. iFogSim[9] is a system that evaluates resource management and scheduling techniques fit for fog settings based on their impact on energy consumption, operational costs, latency, and network congestion. The iFogSim helps programmers weigh the time, money, and effort required to build an app against the available network bandwidth. It builds on the capabilities of the CloudSim simulator and shares many of its features, including the ability to run many applications in parallel and migrate between modules within an application. Applications for IoT, such as real-time stream processing throughout the whole network, can benefit from creating and using iFogSim, a simulator designed specifically for simulating fog settings alongside IoT and Cloud.

D. *MyiFogSim*

MyiFogSim [10] is an add-on to iFogSim that promotes mobility by making it possible to migrate virtual computers on the go. AppModule migration is also implemented. The primary goal of the MyiFogSim simulator is to keep the QoS for mobile customers intact in a fog setting. In addition to inheriting classes from CloudSim and iFogSim, MyiFogSim is being built from the scratch to include its unique implementations of Coordinate, ApDevice, MobileDevice, MobileSensor, MobileActuator, MigrationDecision, MigrationPolicy, MigrationSensor, and MigrationActuator.

E. *FogTorchπ*

FogTorchπ [11] is a novel simulation tool based on a paradigm designed to facilitate the deployment of QoS-aware Internet of Things applications in fog settings. It uses Monte Carlo simulations to enable quality of service in network connections. Its simulation does not support mobility, but its primary purpose is to guarantee quality of service and manage the use of fog resources. It is possible that this Java-based option could be the best option for a speedy and trouble-free deployment evaluation. Because

it is a prototype, the software is not yet complete [156].

F. *EmuFog*

Docker-based software can be emulated with the help of the EmuFog [12] framework, which is designed to work in fog environments. The need for a fog computing testing environment is satisfied. EmuFog, a MaxiNet [56]-based network emulator, is a flexible and scalable platform for analyzing and validating novel routing algorithms in the context of cloud-based data centers. EmuFog allows developers to generate a topology for an expandable fog infrastructure and then deploy fog nodes in two simple steps.

G. *Fogbed*

Fogbed [13] is a framework and toolkit integration created to prototype fog components in rapidly virtualized environments. The resources for developing cloud and fog testbeds are provided. Fogbed allows the deployment of fog nodes as software containers in a wide range of network topologies. Modifying the network topology is as simple as using the Fogbed API to connect, add, or remove containers. FogBed also enables dynamically adjusting the limits placed on a container's CPU and Memory usage.

H. *Yet Another Fog Simulator (YAFS)*

There is a new simulator based on discrete-event called Yet Another Fog Sim (YAFS) [14] that can be used to examine the architecture of software programs. It entails strategies for placing, scheduling, and routing fog networks. YAFS was developed in Python with the PEP8 guideline. To create instances of discrete-event simulations, YAFS makes use of Simpy, a general library. Task creation, calculation, and link transmission events are all automatically logged in a Comma Separated Value (CSV) format by YAFS. Using YAFS, a lot of metrics, such as response time, network delay, network utilization, and waiting time, can be evaluated.

I. *FogNetSim++*

FogNetSim++ [15] is a freely distributable C++-based simulator. It takes its cue from the OMNeT++ [161], a discrete event simulator built for modeling computer networks and other forms of distributed computing. It is an event-driven emulator whose foremost purpose is to deliver static and live fog settings. All currently available OMNeT++ modules can be easily incorporated into FogNetSim++. Researchers can quickly adapt the components of FogNetSim++ to run the simulations. Its primary objective is to improve the sensor support of existing frameworks. However, when modeling latency-sensitive applications, it is crucial to account for network factors such as error and data rates, which they have not done.

J. *MobFogSim*

MobFogSim [16] is an add-on to iFogSim that facilitates the modeling of mobile devices and the transfer of services in fog computing. Numerous classes, such as the Coordinate class, the AppDevice class, and a number of mobile and migration classes, are used to implement mobility and migration functionality. It helps to manage mobile fog services, plan for capacity, and implement network slices. As a result, MobFogSim is an all-encompassing resource for gauging fog computing networks from the perspective of both stationary and mobile users.

K. *Enigma*

Many existing simulators struggle to scale correctly whenever additional components are added to the system. In addition, not all simulators include methods to replicate mobile devices or to track their whereabouts. To address this issue, the gENeric Iot edGe siMulAtor (ENIGMA) [17] is introduced, which provides an API for incorporating mobile device images into graphical maps. This API provides a

graphical overlay showing the temporal distribution of mobile devices inside a user-defined area. It can simulate hundreds or thousands of mobile devices and construct massive infrastructures in edge and fog computing settings.

L. *Dissect-Cf-Fog*

DISSECT-CF-Fog [18] is a simulator that simulates IoT workflow applications in cloud and fog settings. Complex network configurations for data center management and models for IoT devices and services are provided. Large-scale studies involving DISSECT-CF-Fog may be possible, especially if tens of thousands of dynamic entities are present. DISSECT-CF's two main benefits over competing simulation tools are its advanced IaaS stack modeling and integrated resource-sharing approach.

M.*FogDirSim*

The FogDirSim[19] simulator was developed with the CISCO FogDirector API as its foundation. This Python program evaluates and compares several fog computing apps' management rules. It can predict when applications will be available and when they will be unavailable, and it can adjust to variations in application load. It provides a qualitative evaluation of its resistance to fluctuating device failure rates and the convergence rates of various management strategies. FogDirSim is the first prototype that provides a simulation environment that uses the widely adopted RESTful application programming interface to simulate many application management policies based on user-specified parameters like as energy, time, etc.

N. *DockerSim*

DockerSim [20] is a cloud simulation tool that improves upon and expands upon iCanCloud and OMNet++. In addition to comprehensive packet-level routing as well as network protocol behaviors, full operating system-level process management and scheduling behaviors for different levels are provided. It also provides a complete method for modeling application-layer software as a service deployment through various queuing networks.

O. *PureEdgeSim*

PureEdgeSim[21] is a simulation tool that replicates and analyzes real-world cloud, fog, and edge computing environments to determine the effectiveness of numerous resource management strategies. Everything related to modeling and simulating fog and Edge computing is included. CloudSim Plus's discrete-event simulation capabilities are the backbone for the system's inter-part communication. In addition, it successfully models computational activities by utilizing its extensive and easily extendable library that addresses all aspects of cloud computing, including a wide range of resources and services. Because of this, PureEdgeSim only has a minimal set of classes to simulate fog and edge computing settings. It supports several device kinds, mobility, a realistic network model, and other capabilities of usefulness and scalability. PureEdgeSim was made using Java, which is by far the most common programming language.

P. *Edge Fog Simulator*

Margariti et al. [22] developed a distributed cloud architecture called the Edge-Fog cloud to manage the massive and widely dispersed data produced by the IoT. The top qualities of the fog computing setting have been incorporated into this simulator. The system can be broken down into two parts: the edge devices at the network's outskirts and the fog devices in its innermost core. This network of devices is an instance of a distributed system. The Edge-Fog cloud's work allocation methodology reduces deployment times and costs compared to other approaches. The Python programming language was chosen for the development of this simulator.

Q. *FogWorkFlowSim*

FogWorkFlowSim[23] offers an alternate approach to fog computing simulation. It is a Java application with an intuitive GUI for evaluating various project and resource management strategies. The user defines and configures the devices to be used in each of the three layers that make up the fog computing environment. FogWorkFlowSim examines a system's effectiveness with respect to these three metrics: energy use, processing time, and cost.

VII. COMPARATIVE ANALYSIS

A comparative analysis of all discussed simulators has been presented in Table 1. The simulators' source code discussed in Table 1 is available on GitHub. Each simulator is accompanied by a manual outlining its purpose and operation. It is essential to note that the quality and completeness of the documentation vary accordingly. The community's maintenance of the simulator relies heavily on documentation. In addition, this comparative analysis can assist researchers in determining whether the simulator is appropriate for their research or not.

Simulator	**Year**	**Language**	**Base Simulator**	**Pros**	**Cons**
CloudSim [7]	2010	Java	SimJava	Easy to use and scalable simulator for cloud environments	Performance unpredictability
Cloud-Analyst [8]	2010	Java	CloudSim	Support all functionalities of CloudSim and easy-to-support GUI also provides high flexibility.	Resource management policies simulation needs improvement.
iFogSim [9]	2017	Java	CloudSim	Effectively evaluate the resource scheduling and management policies	Node to node communication and mobility is not supported
MyiFogSim [10]	2017	Java	iFogSim	Support mobility and migration of VMs	Does not support fault tolerance
FogTorchπ [11]	2017	Java	FogTorch Prototype application	QoS aware deployment of applications	Does not support mobility of devices
EmuFog [12]	2017	Python	MaxiNet	Simulate large-scale real-world networks	Mobility is not supported
Fogbed [13]	2018	Python	MiniNet	To evaluate real-world problems in the fog	Restricted emulation at

				paradigm	cloud and edge layer
YAFS [14]	2019	Python	NA	Support dynamic deployment of applications in complex networks	Lack of documentation
FogNet Sim++ [15]	2018	C++	OMNet++	Support handover mechanism and simulate large fog networks	VM migration is not supported
MobFog Sim [16]	2020	Java	iFogSim	Mobility of devices is supported.	Complex network configuration
ENIGM A [17]	2022	C++	SimGrid	Simulation of complex and large systems, highly scalable, support mobility	Difficult to use because of complex deployment
DISSEC T-CF-Fog[18]	2020	Java	DISSECT-CF	Evaluate IoT and scientific workflow in cloud and fog environment	Lack of several functionalities
FogDirSi m [19]	2018	Python	FogDirector RESTful API	Effectively evaluate different application management policies	Less support to time-sensitive applications
DockerSi m [20]	2017	C++	OMNet++ & iCanCloud	Efficiently support large-scale and complex simulation scenario	Performance limitations
PureEdg e Sim [21]	2019	Java	CloudSim+	Simulate resource management policies in edge as well as fog environments. Support Mobility and Heterogeneity of devices	VM migration is not supported.
Edge Fog Simulato r[22]	2016	Python	NA	Simulate task and resource allocation in edge and fog environments effectively.	The energy model is not supported.
FogWor kFlow Sim[23]	2019	Java	iFogSim & WorkSim	Simulate complex network topologies and diverse computing	Mobility of End Devices is not considered.

				resources	

Table 1 Total Comparative Analysis of Various Simulators

VIII. CONCLUSION

New approaches in networking and communication settings can be evaluated using simulation. However, a realistic evaluation of these solutions requires a simulator that provides an environment sufficiently close to the one being recreated. Although there are multiple simulators available, each of them has a few properties that set them apart from one another. These features can be used to evaluate various mechanisms in cloud-fog-edge computing paradigms. In this paper, a conceptual review is presented on numerous simulators of the cloud, fog, and edge paradigms. More simulators are planned to be included in the evaluation in upcoming development. It would be interesting to extend the conceptual study in this work to include a practical evaluation of these simulators. The article also highlights the pros and cons of using these simulators, which assists researchers in analyzing the tools for academia as well as industry.

REFERENCES

1. M. Chiang and T. Zhang, 'Fog and IoT: An Overview of Research Opportunities', IEEE Internet Things J., vol. 3, no. 6, pp. 854–864, Dec. 2016, doi: 10.1109/JIOT.2016.2584538.
2. M. Al Masarweh, T. Alwada'n, and W. Afandi, 'Fog Computing, Cloud Computing and IoT Environment: Advanced Broker Management System', J. Sens. Actuator Netw., vol. 11, no. 4, p. 84, 2022.
3. D. Huang and H. Wu, 'Edge clouds—pushing the boundary of mobile clouds', Mob. Cloud Comput., pp. 153–176, 2018.
4. R. Mahmud, R. Kotagiri, and R. Buyya, 'Fog Computing: A Taxonomy, Survey and Future Directions', in Internet of Everything, B. Di Martino, K.-C. Li, L. T. Yang, and A. Esposito, Eds., in Internet of Things. Singapore: Springer Singapore, 2018, pp. 103–130. doi: 10.1007/978-981-10-5861-5_5.
5. P. Hu, S. Dhelim, H. Ning, and T. Qiu, 'Survey on fog computing: architecture, key technologies, applications and open issues', J. Netw. Comput. Appl., vol. 98, pp. 27–42, Nov. 2017, doi: 10.1016/j.jnca.2017.09.002.
6. M. Aazam, M. St-Hilaire, C.-H. Lung, I. Lambadaris, and E.-N. Huh, 'IoT resource estimation challenges and modeling in fog', Fog Comput. Internet Things Intell. Edge, pp. 17–31, 2018.
7. R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, 'CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms', Softw. Pract. Exp., vol. 41, no. 1, pp. 23–50, 2011.
8. B. Wickremasinghe, R. N. Calheiros, and R. Buyya, 'CloudAnalyst: A CloudSim-Based Visual Modeller for Analysing Cloud Computing Environments and Applications', in 2010 24th IEEE International Conference on Advanced Information Networking and Applications, Perth, Australia: IEEE, 2010, pp. 446–452. doi: 10.1109/AINA.2010.32.
9. H. Gupta, A. V. Dastjerdi, S. K. Ghosh, and R. Buyya, 'iFogSim: A Toolkit for Modeling and Simulation of Resource Management Techniques in Internet of Things, Edge and Fog Computing Environments'. arXiv, Jun. 06, 2016. Accessed: Jan. 19, 2023. [Online]. Available: http://arxiv.org/abs/1606.02007

10. M. M. Lopes, W. A. Higashino, M. A. M. Capretz, and L. F. Bittencourt, 'MyiFogSim: A Simulator for Virtual Machine Migration in Fog Computing', in Companion Proceedings of The10th International Conference on Utility and Cloud Computing, in UCC '17 Companion. New York, NY, USA: Association for Computing Machinery, 2017, pp. 47–52. doi: 10.1145/3147234.3148101.
11. A. Brogi, S. Forti, and A. Ibrahim, 'How to Best Deploy Your Fog Applications, Probably', in 2017 IEEE 1st International Conference on Fog and Edge Computing (ICFEC), 2017, pp. 105–114. doi: 10.1109/ICFEC.2017.8.
12. R. Mayer, L. Graser, H. Gupta, E. Saurez, and U. Ramachandran, 'EmuFog: Extensible and scalable emulation of large-scale fog computing infrastructures', in 2017 IEEE Fog World Congress (FWC), 2017, pp. 1–6. doi: 10.1109/FWC.2017.8368525.
13. A. Coutinho, F. Greve, C. Prazeres, and J. Cardoso, 'Fogbed: A rapid-prototyping emulation environment for fog computing', in 2018 IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–7.
14. I. Lera, C. Guerrero, and C. Juiz, 'YAFS: A simulator for IoT scenarios in fog computing', IEEE Access, vol. 7, pp. 91745–91758, 2019.
15. T. Qayyum, A. W. Malik, M. A. K. Khattak, O. Khalid, and S. U. Khan, 'FogNetSim++: A toolkit for modeling and simulation of distributed fog environment', IEEE Access, vol. 6, pp. 63570–63583, 2018.
16. C. Puliafito et al., 'MobFogSim: Simulation of mobility and migration for fog computing', Simul. Model. Pract. Theory, vol. 101, p. 102062, 2020, doi: https://doi.org/10.1016/j.simpat.2019.102062.
17. E. Del-Pozo-Puñal, F. García-Carballeira, and D. Camarmas-Alonso, 'A scalable simulator for cloud, fog and edge computing platforms with mobility support', Future Gener. Comput. Syst., 2023.
18. A. Markus, M. Biro, and A. Kertesz, 'Analysing IoT Applications with DISSECT-CF-Fog in Simulated Fog and Cloud Environments', 2021.
19. S. Forti, A. Pagiaro, and A. Brogi, 'Simulating fogdirector application management', Simul. Model. Pract. Theory, vol. 101, p. 102021, 2020.
20. Z. Nikdel, B. Gao, and S. W. Neville, 'DockerSim: Full-stack simulation of container-based Software-as-a-Service (SaaS) cloud deployments and environments', 2017 IEEE Pac. Rim Conf. Commun. Comput. Signal Process. PACRIM, pp. 1–6, 2017.
21. C. Mechalikh, H. Taktak, and F. Moussa, 'PureEdgeSim: A Simulation Toolkit for Performance Evaluation of Cloud, Fog, and Pure Edge Computing Environments', in 2019 International Conference on High Performance Computing & Simulation (HPCS), Dublin, Ireland: IEEE, Jul. 2019, pp. 700–707. doi: 10.1109/HPCS48598.2019.9188059.
22. A. Kesarwani and P. M. Khilar, 'Development of trust based access control models using fuzzy logic in cloud computing', J. King Saud Univ.-Comput. Inf. Sci., vol. 34, no. 5, pp. 1958–1967, 2022.
23. X. Liu et al., 'FogWorkflowSim: An Automated Simulation Toolkit for Workflow Performance Evaluation in Fog Computing', in Proceedings of the 34th IEEE/ACM International Conference on Automated Software Engineering, in ASE '19. IEEE Press, 2019, pp. 1114–1117. doi: 10.1109/ASE.2019.00115.