IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 22, 2021, accepted May 8, 2021, date of publication May 12, 2021, date of current version May 21, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3079501

CloudSimHypervisor: Modeling and Simulating Network Slicing in Software-Defined Cloud Networks

ANDREWS O. NYANTEH, MAOZHEN LI, MAYSAM F. ABBOD, (Senior Member, IEEE), AND HAMED AL-RAWESHIDY, (Senior Member, IEEE)
Department of Electronics and Computer Engineering, Brunel University London, Uxbridge UB8 3FG, U.K.
Corresponding author: Andrews O. Nyanteh (andrewsoffei.nyanteh@brunel.ac.uk)

ABSTRACT Software-Defined Networking (SDN) is an innovative technology which provides a programmable network control which is decoupled from the physical infrastructure. Network Virtualization (NV) is the phenomenon where a given physical network infrastructure and its resources are abstracted to create multiple logical virtual network slices of the underlying substrate. NV enables independent virtual networks to co-exist on one or more shared physical network infrastructure. Edge computing makes use of the edge resources in close proximity to end-users to reduce service delay and the network traffic volume in the end-to-end networks. Similarly, network slicing which is a key enabling technology for 5G networks is designed to support different services from different platforms at different scales enables sharing of physical network infrastructure on many different virtual network layers. These innovative technologies and strategies have gained significant attention from both academia and industry as they have the potential to maximize network resource utilization and optimize end–to–end network service delivery in 5G solutions deployment. To enable continuous simulation and development of applicable 5G networking concepts using these technologies, there is a need for an accessible and easy-to-learn testbed which is able to efficiently measure the performance of physical and virtual network capacities, provisioning approaches and management of multiple architectural models using large-scale network slicing configurations in a repeatable and controllable manner. These tools and toolkits provide scalable, lightweight and controlled cloud simulation environments necessary to analyse network traffic flows, allocation capacities and policies and the behaviour of multiple heterogeneous networks at an extremely low cost as compared to the huge financial commitments involved in conducting similar experiments in a real-life event. Existing solutions do not support Network Slicing and end- to -end heterogenous network automation which are key enablers of 5G network implementation. Hence in this paper, the CloudSimHypervisor framework is developed in this based on CloudSimSDN-NFV. The complete architecture and features of the CloudSimHypervisor framework and some used cases are presented in this paper. We validate the CloudSimHypervisor framework with two use case experiments in the cloud computing environment: Joint compute and network resource utilization and network traffic prioritization. Results from these experiments display the efficiency of the CloudSimHypervisor in estimating and measuring processing speed, transmission speed, compute and network usage efficiency and energy consumption.

INDEX TERMS Virtual software-defined network, openflow, network virtualization hypervisor, tenant controllers, CloudSimHypervisor.

I. INTRODUCTION

The emergence of modern technologies such as Cloud Computing, Software Defined Networking (SDN), Network Functions Virtualization (NFV), Edge Computing and the Internet of Things (IoT) have rapidly transformed end-to-end service provisioning over the past decade [1] and have smart services for subscription. On -demand cloud user requests with well-defined Service Level Agreements (SLA) often have different Quality of Service (QoS) requirements

The associate editor coordinating the review of this manuscript and approving it for publication was Tariq Umer.

for processing and delivery. Software defined networking (SDN) has emerged as potent networking technology which makes control of communications network flexible. SDN has the capability to separate the control function from the forwarding function of the networking devices in the physical infrastructure and defines open and programmable interface with the use of OpenFlow (OF) protocol [2]. SDN presents a programmable centralized network control unit known as controller which gives a global view of the entire network [3].

This enables dynamic configuration of the network control logic and management, allowing components in the infrastructure plane (physical network) to adjust quickly to change in requests and workload requirements on-demand [4], [2]. The centralized network control of SDN operates in collaboration with some application programming interfaces (APIs) to enable communication in the entire network. Network Virtualization (NV) is the phenomenon where a given physical network infrastructure and its resources are abstracted to create multiple logical virtual network slices of the underlying substrate [5]. NV enables independent virtual networks to co-exist on one or more shared physical network infrastructure [4]. Each of these virtual networks has specific tailored network service demands or end-user applications requirements from the underlying physical network substrate [6]. Network service providers deliver services to different service users via sharing of these independent virtual networks deployed over a shared physical substrate. Network Virtualization is considered an enabling networking technology for the next generation internet. It presents the potential to overcome the current network ossification and limitations in the current internet and communication networks by enhancing network resource efficiency and resource sharing [7]. These two technologies though independent can complement each other to leverage the combined advantage of their concepts to enhance the operational efficiency of network and communications systems.

With the use of OpenFlow [8] protocol multiple virtual software defined networks (vSDNs) also known as tenant networks of a given physical SDN infrastructure can be executed on a network virtualization hypervisor. Network slicing may be defined as the logical partitioning of a physical network into independent virtual networks which enables the multi-tenancy paradigm [9].Network slicing is identified as a key enabling technology to enable 5G networks with multi-services. Network slicing enables a transition from a *network-as-an-infrastructure* setup to a *network- as-a-service* which allows numerous 5G smart services with diverse requirements [10], [11]. It is mainly enabled by NFV, SDN, cloud computing, and edge computing [12].

These 5G networks would be required to support different types of services from different industries at different scales and enable sharing of physical network infrastructures but on many different virtual network layers. This implies that, end-to-end communication paths would have to traverse multiple autonomous systems and layers operated by different organizations each with different network policies and hence incurring high network traffic latency, high network overhead and a higher probability of violating Service Level Agreements (SLAs).

Although cloud computing enables the deployment of service-oriented network architecture, large-scale cloud network infrastructures and datacentres in different geographical locations, end-to-end service provisioning for user demands in these heterogeneous network domains has proven to be very difficult to achieve. This is because there's no uniform inter-domain service delivery platform with standardized operational policies for service providers to effectively provision end-to-end network services. There have been quite a number of research studies in recent years to investigate and implement network architecture to mitigate these challenges by defining network topologies which comprise heterogenous network resources with accurate specifications [5], [8]–[11]. However, the tools which are available for these studies are limited in resource capacity and specifications to implement and experiment network models and techniques. For instance, there are a number of proof-of-concept platforms which cannot be used for large-scale end-to-end deployments because it is extremely expensive and challenging to setup proposed multiple heterogeneous network infrastructure and efficiently evaluate the varying performance metrics required in experiments in the real world.

Network setups and topology of OpenFlow switches (hosts) can be emulated using Mininet [13]. Primarily, Mininet provides a development environment and a virtual testbed to create prototypes of a variety of networks which includes software defined networks. It also has the capability to perform experiments involving discrete load balancing and network traffic management policies in the controller of software defined networks. However, Mininet does not have the capability to perform cloud network resource management procedures such as virtualizing network resource, network resource consolidation, network slicing and VM placement. Therefore, there is a need for a cloud environment simulator, which has a user friendly and easy-to-learn testbed environment and is able to measure the performance metrics and assess the behaviours and varying large-scale infrastructure demands of modern networking technologies in a controlled environment.

In this paper, the CloudSimHypervisor is presented as a cloud computing-based simulation framework. The CloudSimHypervisor has the capability to simulate network slicing and its multiple heterogenous network architectures, as well as large-scale end-to-end networks automation with the use of the network virtualization hypervisor.

The main contributions that we present in this paper are;

- Designing the architecture and logic for simulating Network Slicing using SDN, NFV and edge compute resources.
- Modelling the Network Virtualization Hypervisor which supports multiple heterogeneous network slices.

- Validating the efficiency of CloudSimHypervisor in comparison with the existing CloudSimSDN-NFV simulation framework.
- Evaluating the accuracy of the CloudSimHypervisor simulation framework in measuring end- to -end network performance and energy conservation with use case scenarios.

The remainder of the paper is organized as follows. In Section II, we outline other research works which are related to this simulator as well as highlighting the peculiar functions CloudSimHypervisor. Section III, we emphasize the requirements of the simulation, then Section IV provides the description of overall framework design and its components in detail. The validation process of the simulator is explained in Section V, followed by an evaluation with use case scenarios for three-tier applications in Section VI. Section VII summarizes the outcomes of this paper, VIII outlines the conclusions of all the observations and analysis and Section IX details future work and direction.

II. RELATED WORK

Many different research studies have contributed to methods to simulate and emulate of cloud, SDN, NFV to literature. Other proof-of-concept setups provide the platform to measure various performance metrics of IoT networks and edge compute. In this section, we review some of these research studies which are related to the CloudSimHypervisor. Núñez, *et al.* in [14] proposed iCanCloud simulator which simulates large scale cloud experiments with specific regards to enabling a cost-performance analysis of workloads which are executed in a cloud datacentre. iCanCloud is equipped with the INET framework Network which allows simulation of network infrastructures which includes network devices e.g. (routers and switches) and protocols such as User Datagram Protocol (UDP) and Transmission Control Protocol (TCP). This application, however, does not support the modelling and simulation of SDN controllers and related features. Stanford University developed Mininet to emulate SDN controllers in a single machine running Linux operating system (OS). As an emulation application, Mininet can emulate SDN functions with real-world scenarios with any OpenFlow- compatible SDN controller [1].

The network virtualization functions provided by the control groups and namespaces of the Linux kennel virtually creates network hosts and switches. Although the use of network drivers in Linux provides explicit empirical results from SDN control logic implementation, limitations in resource usage and capabilities of the Linux OS limits cloud-scale (thousands of machines) during simulations. Calheiros, et al in [9] presented CloudSim is a discrete event-based cloud environment simulator which is implemented in Java IDE. This application simulates interactions and events of components and processes of datacentre networks for instance, workloads scheduling, VM placements and resource provisioning [1]. Results of simulations from CloudSim are computed based on time for sending and receiving events between cloud entities such as broker, host and cloud datacentre [9].

CloudSim, makes provision for its entities to be extended. Thus, it allows flexibility for new elements to be developed and implemented with existing entities and events. This has led to the development of projects such as CloudSim Plus, CloudSimSDN, ContainerCloudSim and CloudSim SDN-NFV. However, CloudSim does not support the programmable control and network performance evaluation of SDN, the virtualization and orchestration functions of NFV and the capabilities of Edge Compute to make provision of network resources to brought in proximity to end users. Garg and Buyya in [3], suggested NetworkCloudSim framework, an extension of CloudSim which makes provision for simulating generalised application models and scalable networks with communication tasks such as message passing application and workloads in a datacentre. Network elements such as switches and links which are not available in CloudSim are implemented in NetworkCloudSim. These elements are used to analyse and estimate network transmission time. Although NetworkCloudSim simulates scalable networks in a datacentre, it cannot simulate NFV functions and orchestration and edge compute.

Aslanpour *et al.* [15] introduced AutoScaleSim, an application which extends CloudSim simulator. This application supports auto-scaling of Web applications in cloud environments in a bespoke and scalable manner. AutoScaleSim facilitates efficient implementation and evaluation of auto-scaling strategies for monitoring, analysing, planning and execution.

CloudSimSDN-NFV simulation framework was proposed bySon and Buyya in [1]. This simulation framework provides support for NFV functions and orchestration and edge compute in a cloud computing environment. It extends CloudSimSDN [16] which simulates SDN-enabled clouds to include NFV functions in edge-clouds to enable support for edge and cloud datacentres. It has capabilities for auto-scaling policies for service function chaining (SFC) and virtual network function (VNF) placement policies in an integrated edge and cloud computing environments. It also supports VNF allocation, and migration and key edge computing concepts such as differentiated datacenter capacities. CloudSimSDN-NFV has attracted attention from both academia and industry as it focuses on Simulating SDN enabled clouds, edge-clouds and inter-datacenter networks. However, it does not support network slicing which is a significant concept in enabling end-to-end network automation in 5G networks.

Chaabnia et al in [17] proposed a proof-of-concept hierarchical slicing model for IoT-based smart homes which comprise two layers, the control plane slicing and the home gateway slicing. This slicing architecture was designed to support only one vSwitch for the entire smart home network. The control plane slicing utilized flowvisor to distribute address schemes. Application and network traffic as well as bandwidth allocation are manually classified based on data

rates. This work adopts Mininet as a test bed for experimenting and evaluating proposed strategies.

The existing toolkits which have been discussed in this section are similar to the work which is proposed in this paper. However, the components and apparatus required for these simulators and emulators are very expensive and it can be time consuming setting them up to experiment proposed strategies. Furthermore, they do not support simulating and experimenting large-scale networking architectures which are required to efficiently evaluate and validate proposed concepts for 5G technologies. Hence the need for an efficient, scalable and easy to use toolkit which has the capacity to support the behaviours of large-scale multiple heterogeneous at extremely low cost. This paper presents a simulation framework for modelling and simulating network slicing and heterogeneous network tenancy and their application in evaluating performance and strategies in end-to-end 5G networks.

III. EVENT-DRIVEN SIMULATION

Simulations in CloudSim follow a well-defined process of sending and receiving events between entities. For instance, to create a virtual machine on a host in a datacentre, a VM request event is sent to the datacentre entity. The datacentre entity receives the event and allocates resources to execute the requested VM through the use of VM allocation policies assigned to it. Likewise, the VM, through its processing scheduler can receive a CPU workload event through a datacentre entity. The processing scheduler in this activity receives the CPU workload when it arrives, calculates its processing end time and returns the completed event with the end time. This procedure is applied in CloudSimHypervisor as well to simulate network transmissions, creating and deleting vSDNs, VNF and inter-networks events. Simulation events are sent and received among entities while the policies and schedulers calculate event delays. These policies and schedulers can be customised and reused to implement various scenarios and algorithms [1].

A. SOFTWARE-DEFINED CLOUD NETWORKS SIMULATION

The effectiveness of new strategies, policies and algorithms can be estimated by adopting methods to experiment them on a large-scale. This is because it may not be practically possible to do the evaluate these new policies and algorithms as the outcome is unpredictable. Alternatively, practical testbed experiments can be conducted to validate and evaluate new strategies and algorithms on a small-scale to however, the outcome of such experiments may not be as accurate as that of large-scale experiments. Empirical implementation which can also be employed to validate new strategies, policies and algorithms can be time consuming and expensive. Hence, the adoption of simulation in science with which multiple experiments with different parameters and configuration can be conducted mostly with automated scripts in a short time.

Simulation also simplifies the evaluation of algorithms, policies and strategies with reasonable accuracy and be compared with baselines under same conditions. Similarly, simulating network slicing and its multiple heterogeneous networks in an SDN, NFV and edge compute environment to reduce the time for evaluating and validating the policies, strategies and processes involved. With simulation, experiments are reproducible in controlled environments [18]. Furthermore, simulation toolkits allow for different scenarios to be implemented and evaluate to discover enhanced novel concepts in a short space of time. A toolkit which has the capability to support designs and experimentation of Network Slicing in an SDN, NFV and Edge computing environment to evaluate different Quality of Service (QoS), metrics under different experimental conditions is required. To achieve the stated objectives, the experiments performed had the following requirements:

- The capability to simulate SDN physical infrastructure, components and their configuration.
- Support to simulate flows and different physical and virtual network allocation policies which are implemented per flow in an integrated cloud network solution.
- The capability to model and simulate Network Slicing of the physical SDN and its behaviours.
- Support a model the Network Virtualization Hypervisor, by integrating the control attributes of SDN and the virtualization and slicing attributes of NFV.
- Support to model the co-existence of multiple heterogeneous virtual networks on the Network Virtualization Hypervisor.
- The ability to model isolated independent virtual SDN networks (vSDNs).
- Support for network specific tenant controllers for each virtual software defined network (vSDN).
- Provision for policies which support network slice selection, slice traffic allocation and management, placement algorithms.
- Support to model Network Functions Virtualization, NFV.
- The capability to represent dynamic virtual networks requests (VNR), cloud workloads and their properties.
- Support for virtual network embedding (VNE).
- Support for multiple heterogeneous networks design and configuration.
- The capacity to simulate edge compute, and its behaviours.
- Capacity to evaluate performance of the above-mentioned frameworks with use case scenarios.

The above mentioned requirements and others motivated the design and development of the CloudSimHypervisor simulation framework. The details of this framework are presented in the next section.

B. CloudSimHypervisor

CloudSimHypervisor was developed by extending CloudSimSDN-NFV [1] which was developed as an extension of CloudSimSDN simulation toolkit [16] which also extended CloudSim simulation toolkit [9]. The details of the

architecture implementing network slicing in SDN, NFV and edge compute is shown in Figure 1.

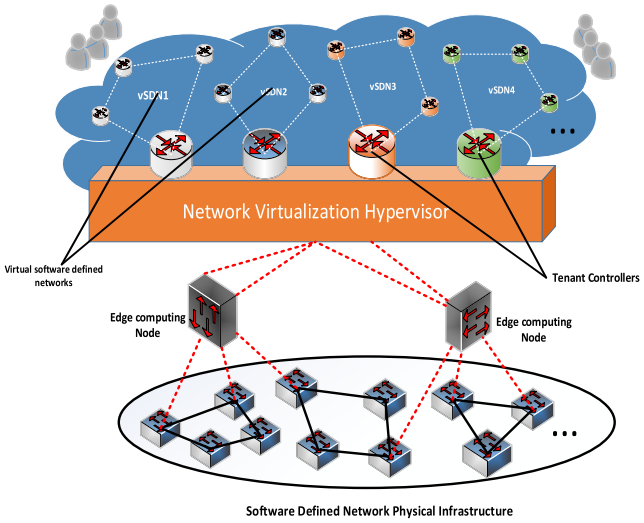


FIGURE 1. The architecture of network virtualization hypervisor in a software defined cloud network environment.

The diagram contains the components that the CloudSimHypervisor framework is made of and employs for simulations. Network slicing is a logical partitioning of a physical network into independent virtual networks which enables multi-tenancy of heterogeneous networks which dynamically allocates resources based on end- user demands. End-user demands may be fixed, or variable based on factors such the type of user (e.g., premium user, standard user or ordinary user), the type of user request (e.g., single application request or multiple applications request), user SLA and location. Each vSDN represents an independent isolated network slice.

The vSDN comprise a slice of the physical infrastructure of the SDN (nodes) and a slice of the programmable network control (tenant controller) as explained in Table 1. The nodes and the tenant controllers are stitched together for the isolated vSDN by the virtualization functions of NFV. The independent tenant controllers may have different network operating systems (NOS).

The network virtualization hypervisor layer creates the network slices, isolates the heterogeneous networks and manages the resource provisioning process. It is also responsible for virtual network embedding (VNE), the phenomenon of mapping resource demands of virtual network requests (VNR) to physical network host components which have adequate resources to execute the various end-user requests. It comprises an integration of the programmable control property of SDN and the virtualization and slicing functions of NFV in a single layer. Edge computing nodes connect physical networks in locations close to end-users, datacenters and edge clouds to the Network virtualization hypervisor. It also sends and receives requests and reallocations from these locations. Our simulation framework simulates

TABLE 1. Comparing NetworkCloudSim, CloudSimSDN-NFV and CloudSimHypervisor.

NetworkCloudSim	CloudSimSDN-NFV	CloudSimHypervisor
Does not support NFV and virtualized cloud resources [3] [8]	NFV is implemented as an independent entity in an edge cloud environment [1]	NFV is integrated with SDN in a unified architecture in a cloud and edge cloud environments
Does not Support Network Slicing [3] [8]	Does not support Network Slicing [1] [19]	Framework introduces the network virtualization hypervisor which creates multiple heterogeneous network slices (vSDNs) to the CloudSim family of simulations framework
User requests are received and processed in VMs. They are not cloud based [3] [8]	User requests are randomly placed in the cloud and are represented basically by VMs [3] [19] [1]	User requests are received by the network virtualization hypervisor and comprise of SDNVMs, network channels
Framework does not support virtualized cloud resources [3] [8]	Framework does not make provision for independent virtual networks (vSDN) [1]	Framework introduces independent virtual software defined networks (vSDNs) to the CloudSim family of applications
Framework does not support virtualized cloud resources and Network Operating System (NOS) features [3] [8]	Framework does not make provision for slicing SDN controller into multiple tenant controllers [19][1]	Framework introduces slicing SDN controller into multiple virtual tenant controllers for each independent vSDN

components in the physical infrastructure of SDN such as switches (core, aggregate, and edge), SDNhosts, physical network links, switching components which are required to create a backbone connection to SDN network resources in multiple cloud datacentres such as gateway switches and intercloud switches, virtual network topologies, a variety of centralized and distributed network control mechanisms (Network operating systems), NFV enabled network service functions to compute varying predefined and dynamically changing Quality of Service (QoS) attributes, and the network virtualization hypervisor.

IV. FRAMEWORK DESIGN

The CloudSimHypervisor framework is able to measure the performance of the heterogeneous network slices such as energy savings, cost efficiency and environmental conservation. Although Network-as-a-Service (NaaS) which involves a merger of software-defined networks (SDN) and network functions virtualization (NFV) as well service-oriented architectures (SOA) is one of the key strategies for the fifth generation (5G) internet, little consideration has been given to simulating integrated network architectures involving all of its key networking technologies. CloudSimHypervisor enables researchers to implement, test and evaluate strategies and algorithms for maximizing network resource utilization

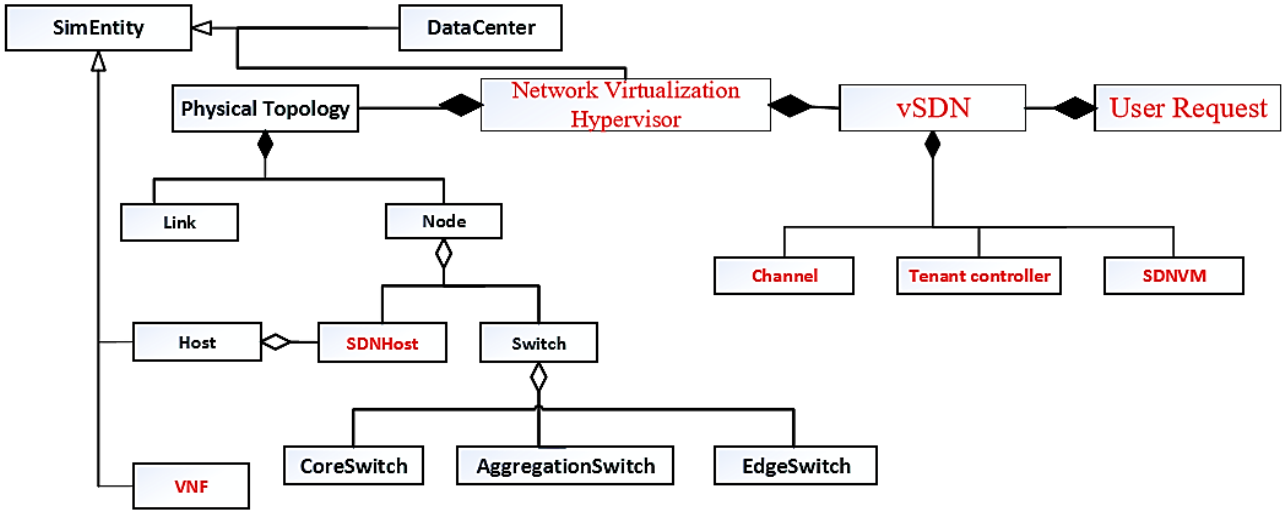


FIGURE 2. Class diagram of CloudSimHypervisor. Note: The classes in red font are new classes added in CloudSimHypervisor.

with the use of the functions and resources it provides for Network Virtualization, Network Slicing and multiple heterogeneous networks.

The concept of virtualizing Software Defined Networking stems from computer resource virtualization and thus network resources of a single or multiple physical SDN infrastructure(s) are sliced and shared by multiple independent vSDNs. This CloudSimHypervisor enables researchers to model network slicing and resources sharing of physical infrastructure of SDN as an independent layer implemented between the physical SDN infrastructure (data plane) and the control plane.

CloudSimHypervisor simulation framework is built on CloudSim [9] a discrete event-driven simulation framework which is designed and implemented in Java IDE. Our simulation framework is designed along the object-oriented programming model just like CloudSim. Peripherals and components of the Network Virtualization Hypervisor and Network Slicing have been designed and developed as Java Classes based on object-oriented setup by extending several Classes in CloudSimSDN-NFV and developing some new Classes as well based on the requirements of the new scenarios implemented in CloudSimHypervisor. This Simulation framework employs the capacity of CloudSimSDN-NFV to model the physical and virtual network infrastructure of SDN, their components and dynamic configuration, setting up virtual network functions (NFV), edge computer nodes and service functions chaining, SFC using its simulation engine.

A. THE CORE LOGIC OF CloudSimHypervisor

The core logic of CloudSim simulates the fundamental compute elements of the cloud infrastructure. Physical hosts in CloudSim are defined with specific settings. The VMs which are hosted by these physical hosts must meet well defined requirements of CPU power, memory and storage size [16]. CloudSimSDN extended components such as datacentre, physical host, VM, VM scheduler, workload scheduler in CloudSim. CloudsimSDN-NFV introduced network functions virtualization (NFV), Service Functions Chaining (SFC) and edge computing resources [1] which makes it possible to simulate heterogeneous datacentres in different geographic domains. However, CloudsimSDN-NFV does not make provision for simulating and deploying a network virtualization hypervisor as a virtualization layer which supports abstraction of hosts and link resources to a composite unit to reuse in setting up independent vSDNs. Figure 2 is a class diagram which illustrates core components of the CloudSimHypervisor.

B. NETWORK VIRTUALIZATION HYPERVISOR

The network virtualization hypervisor is designed to integrate functions of SDN, NFV, SFC and several customizable policies. It is implemented by extending the Network Operating System (NOS), Virtualized Network Function (VNF), Service Functions Chaining (SFC) Classes, packets schedulers and other customisable policies from CloudSimSDN-NFV as shown in Figure 3. As a multi- purpose mechanism which manages the programmable network control, virtualizing and slicing heterogeneous networks, the network hypervisor is modelled to implement:

1. Functions and behaviours supported by Software Defined Network control. Which creates dedicated channels for specific traffic flows and monitors all network channels. It has the capability to calculate the estimated arrival time of packets based on the allocated bandwidth for each channel and the number of packets sharing the channels. Where there are more virtual channels sharing a physical link, each channel size is also included in the bandwidth calculation. The Network Virtualization Hypervisor is also designed to setup forwarding rules and network behaviours which

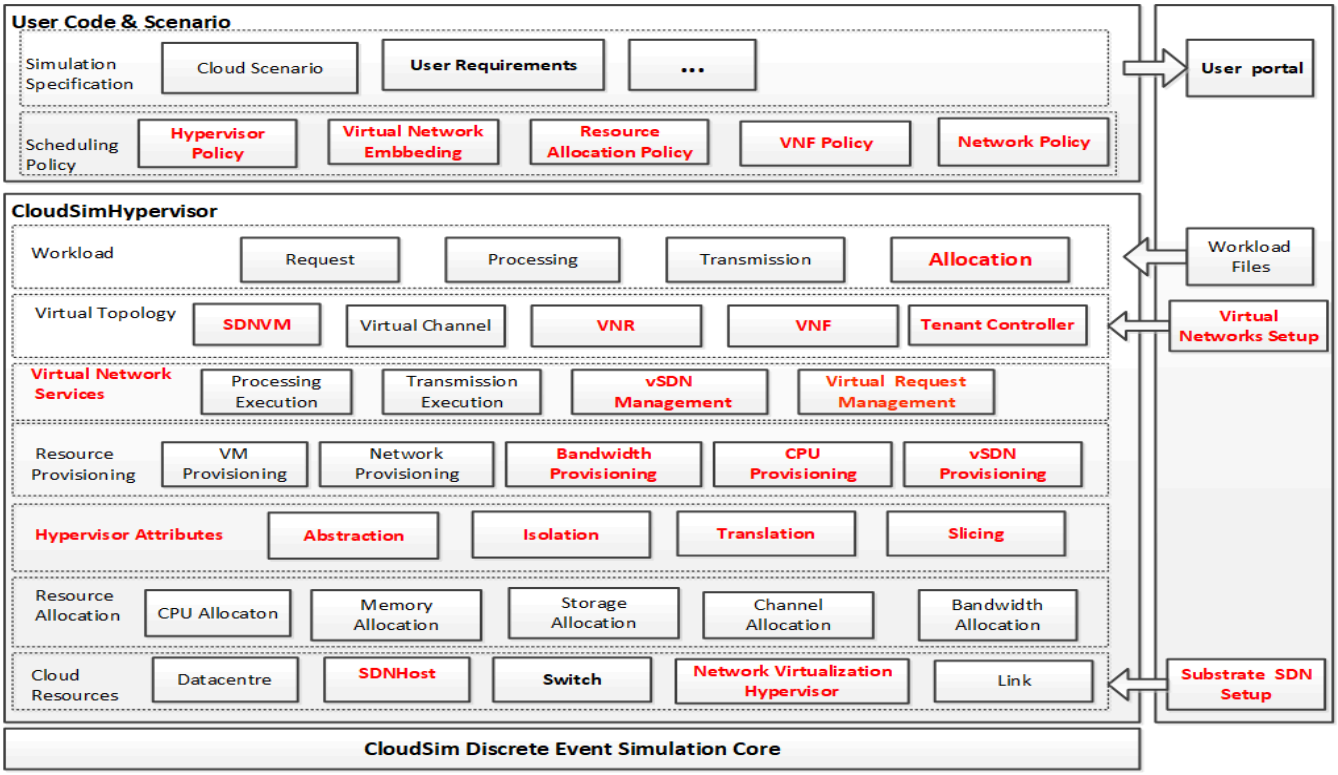


FIGURE 3. CloudSimHypervisor framework design. Note: The fields in red font are new fields added in CloudSimHypervisor.

can be dynamically changed based on the network traffic. The Class makes provision for lists of physical network elements, such as switches and hosts, along with physical links connecting these elements.

2. Functions of Virtualize Networks (VN) and Network Slices. These modules of the network hypervisor have high CPU, memory and storage requirements. The network slices have attributes such as processing capacity which are simulated as the number of cores (Processing Elements) and MIPS (Million Instructions Per Secon). This Class has a defined field for MIPO (Million Instructions Per Operation), which computes throughput of the Network Slices. MIPO defines the length of CPU workload for every network operation within a network slice. MIPO and MIPS have a direct correlation on the volume of network requests which the network virtualization hypervisor allocates to specific network slices. For instance, a network slice which has a processing capacity of 1000 MIPS and 10 MIPO can execute 100 requests (operations) per second throughput.
3. Functions of Virtualize Networks (VN) and Network Slices. These modules of the network hypervisor have high CPU, memory and storage requirements. The network slices have attributes such as processing capacity which are simulated as the number of cores (Processing Elements) and MIPS (Million Instructions Per Secon). This Class has a defined field for MIPO (Million Instructions Per Operation), which computes throughput of the Network Slices. MIPO defines the length of CPU workload for every network operation within a network slice. MIPO and MIPS have a direct correlation on the volume of network requests which the network virtualization hypervisor allocates to specific network slices. For instance, a network slice which has a processing capacity of 1000 MIPS and 10 MIPO can execute 100 requests (operations) per second throughput.
4. Functions for Network Slice Isolation. The ***ServiceFunctionAutoScaler*** and the ***ServiceFunctionForwarder*** policies are leveraged in the Service Functions Chaining (SFC) Class in CloudSimSDN-NFV and extended the Class to support the network slice isolation capability in CloudSimHypervisor. The ***NetworkResourceAutoScaler*** was implemented to increase or decrease the capacity of network resources in a vertical and/or horizontal scaling based on pre-defined auto-scaling policies during simulations. It scales vertically when it increases of decreases the capacity network resources such as bandwidth allocated to network slices and horizontally when it increases or decreases the number of compute resources managed by the slice. It also has the capacity to measure the average end-to-end latencies across

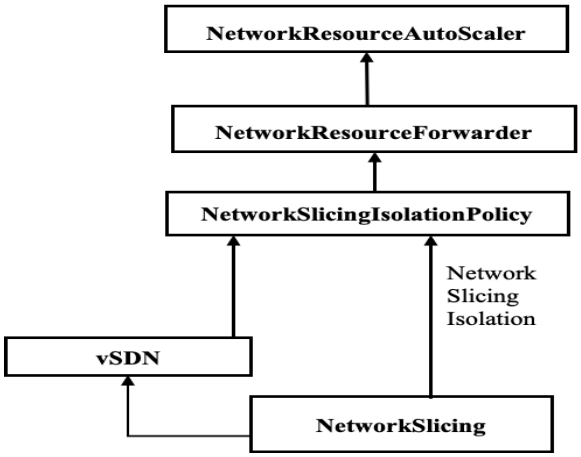


FIGURE 4. Class diagram for implementing network slicing and isolation.

the various heterogeneous network slices managed by the network virtualization hypervisor. The ***NetworkResourceForwarde*** forwards packets to specified isolated network slices during simulations. The forwarder confirms the configurations of the various infrastructures on the heterogeneous network slices with the ***NetworkSliceIsolationPolicy*** and enforces multiple traffic associated with the policy. Figure 4 is a class diagram which illustrates network sling and isolation flow of CloudSimHypervisor.

5. The VmAllocationPolicy is extended which was defined in CloudSim and the HostSelectionPolicy which was modelled in CloudSimSDN-NFV into componentized customizable network resource allocation and network mapping policie. ***VirtualNetworkMappe***r Class was extended to be responsible for mapping the requested virtual network topology onto the physical network elements in the Virtual Network Embedding (VNE) phenomenon. The ***LinkSelectionPolic***y Clas was extended to implement strategies to randomly select one or more links from a list multiple of links in the physical network topology for network resource allocation and VNE regardless of the network capacity. Similarly, Other link selection related policies with practical applications were also implemented. For instance, ***LinkSelectionPolicyDestination***- Address assigns defined links calculated by a modulo operation to specific network destination addresses. ***LinkSelectionPolicyFlowCapacit*** scans the links in the candidate list for the links which are occupied and returns. a list of unoccupied links to use during simulations. This enables the network hypervisor to evenly distribute network transmissions along multiple paths.

C. PACKET SCHEDULER

The packet scheduler is planned in a similar to the Cloudlet scheduling in the CloudSim and CloudsimSDN-NFV. In CloudSim, Cloudlet which has the length of the processing workload models computing workload for processing CPU. CloudletScheduler simulates scheduling of the processing workload in each VM based on the simulation scenario. In CloudletSchedulerTimeShared, the processor capacity is evenly shared by all Cloudlets submitted and currently processed at the VM. For example, if five Cloudlets are submitted to a VM, the CPU capacity of the VM is shared among them so that each Cloudlet can be assigned 20% of the total CPU capacity [1]. On the other hand, CloudletSchedulerSpaceShared processes only one Cloudlet at each time so that 100% of the CPU capacity will be assigned to the first Cloudlet submitted to the VM. The other Cloudlets are in the waiting list and processed in the queue once the earlier Cloudlet completes the processing [1].

In CloudSimHypervisor, the network packet scheduler is designed with Packet and PacketScheduler Classes similar to Cloudlet and CloudletScheduler. Packet Class represents the network transmission workload which has the size of the network packet. PacketScheduler distributes the available network bandwidth among currently transferring Packets with the same source and destination VMs. If multiple flows share the same physical link, the bandwidth of the physical link is distributed among these flows and then PacketScheduler can allocate the distributed bandwidth onto Packets in the scheduler. Similar to CloudletScheduler, we implement PacketScheduler with two models, timesharing and spacesharing. In time-sharing, the available bandwidth is equally shared among Packets from the same source VM to the same destination VM. In SpaceShared, the entire bandwidth of the virtual network is allocated to the first Packet submitted to the network, and the rest are waiting in the queue until the transmission of the first Packet is completed.

D. CALCULATING PACKET TRANSMISSION TIME

Simulation of network requires that the transmission time for data transferred between hosts is calculated. Calculation is straightforward if the data is transmitted for one hop that is not shared with other hosts. However, it is more complicated to estimate travel time when the packet needs to be transferred to the host via multiple hops where some are shared by other hosts. In fact, data is fragmented into several packets involved in multiple fragmentation process on each network layer depending on protocols. The fragmentation processes are complicated and varied on different protocols. Therefore, the transmission process model is simplified and so is the estimation of transmission time. We introduce the class Channel, an end-to-end edge from sender to receiver consisting of multiple links. It is a path for data packets that are going through a series of queues of ports in different switches. The class Link is a physical medium between ports of switches or hosts. The class Transmission refers the transferring data between two hosts which travels through the channel. In each link, bandwidth is first allocated to the priority channel if SDN is configured to allocate a specific amount of bandwidth to the channel. Afterwards, the remaining bandwidth is equally shared by the channels passing through the link.

Thus, allocated bandwidth $BW_{c,l}$ for a channel c in the link l is defined as

$$BW_{c,l} = \frac{BW_l}{N_l} \quad (1)$$

where the link (l) has available bandwidth (BW_l) shared by the number of channels (N_l) [19]. As a channel is composed of multiple links, the transmission speed of the channel basically depends on the least bandwidth among the links. Even if some links have higher bandwidth, there would be a bottleneck when packets pass through a link with lower bandwidth. Thus, for the time period Δt, when no channel has been added or removed, the amount of data Dc transferred from sender to receiver on a channel c can be calculated using Equation 2.

$$Dc = \Delta t \times \text{Min}\,(BW_{c,l}) \quad (2)$$

When a new channel is added, Network Operating System informs all links where the new channel passes through, and existing channels are updated with a new lower bandwidth value. Channels and links are also updated when a data transmission is finished, and the allocated channel is deleted. In this case, the remaining channels will have more bandwidth as there is one less channel using the link. Updated bandwidth values are used to calculate the size of data transferred for the next time period.

E. ABSTRACTING USER REQUESTS

In real world network events, jobs associated with network transport can be abstracted as a combination of computation processing interspersed with transfers of packets. Considering a web service model for instance, when a request is received at the front-end server, e.g. web server, the front-end server computes the request and creates another request to the mid-tier server, e.g. application server. Similarly, the mid-tier servers process the requests receive requests, it receives and transfers them to the back-end server, e.g. database server. Hence, to implement a request which contains both workloads and network transmissions, three classes are implemented: Request, Processing and Transmission where Processing and Transmission Classes are implemented with respect to the Activity interface [19]. Every Request is made up of a list of multiple Activity objects, which are implemented as Process computation or Transmission. Process computations comprise workload (Cloudlet), and Transmission has a network transmission requirement (Package).

Network request is made up of Processing and Transmission objects which usually appear in a well-defined order. For ease of use, a list of requests can be generated in a CSV format which has multiple pairs of Processing and Transmission. To logically estimate network transfer time for each packet, we made provision for Queue in nodes for each flow. For example, if a flow is set up between two hosts, the queue should be set up in the sender's host as well as in all switches that packets go through.

V. VALIDATION

In order to validate the CloudSimHypervisor, we conducted a series of experiments that compared CloudSimHypervisor with CloudsimSDN-NFV with the same workloads. CloudsimSDN-NFV is a simulation application which makes provision for simulating SDN, NFV, SFC and edge computing resources, which considers the execution of cloud applications and resource mainly on the basis of resource migration and placement of VMs in a cloud computing environment. It does not support Network Slicing and end- to -end heterogenous network automation which are key enablers of 5G network implementation. The Network Virtualization Hypervisor possesses the capabilities to simulate Network Slicing and isolated independent heterogeneous virtual networks. Our objective was first to model different Network Slicing scenarios with different data sizes, testbed environment configurations and different shortest paths between the hosts and other network elements. We then analysed how close the data transfer rate between the network elements (host, links, and channels) in CloudSimSDN-NFV and CloudSimHypervisor validate the accuracy of the CloudSimHypervisor. Figure 5 is a diagram which shows a three-tier network topology architecture which was implemented in the experiment.

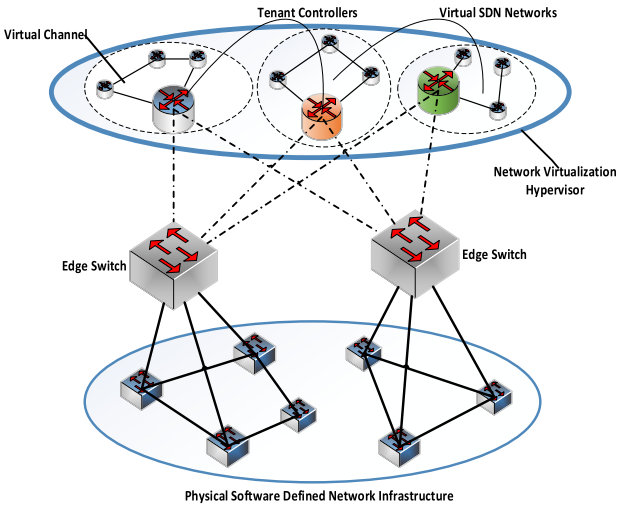


FIGURE 5. Experimental setup for validating CloudSimHypervisor.

A. CloudsimSDN-NFV SETUP

Experimental setup for CloudSimSDN-NFV was done in a Java IDE using the CloudSimSDN-NFV framework [1]. The physical network topology is setup in CloudSimSDN-NFV by creating and adding the physical host, switches (Core, Aggregate, Edge) and links which form the SDN-enabled cloud datacentre in a JSON file. The virtual network topology is the resource deployment request. When network users send VM formation request to the cloud, network resources present the topology of the virtual network with QoS requirements and Service Level Agreements, SLA. This was input as a

JSON file. Workloads of network transmission and compute processing are passed from network users to VMs when the VMs are created in the SDN-enabled cloud datacentres.

In this experiment, VM is placed in each physical host in CloudSimSDN-NFV. Hence, each VM represents a physical machine. Link speed between core and edge switches, and between edge switches and hosts which was set to static and sometimes varied during the experiment.

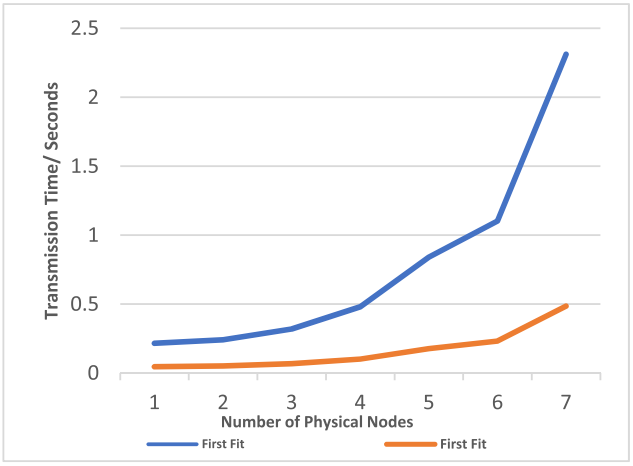
B. TESTBED CONFIGURATION

Three-tier network topology setup which comprises two physical software defined networks with one of them having four hosts and the other three hosts. The setup also has two edge switches which connect the physical software defined network to the network virtualization hypervisor. The hypervisor supports three isolate virtual software defined networks (vSDNs) each representing an independent Network Slice. Each of the vSDN comprise three virtual nodes (SDNVM) and a network specific tenant controller. The bandwidth allocated to the links from the edge switches to the hypervisor and between the edge switch and the hosts in the physical networks are same as used for the setup in CloudSimSDN-NFV. This topology can support a number of scenarios, for instance, transferring data through dedicated routes and transferring data across random links or network elements.

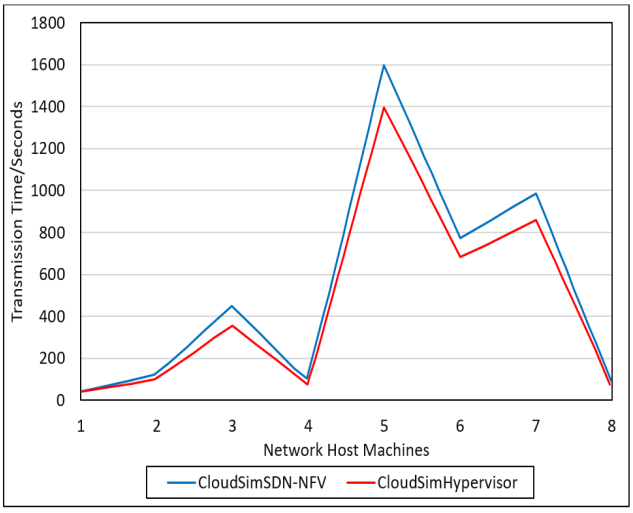
Each host physical network is configured to receive request for compute and network resources and send data with different sizes randomly to the network virtualization hypervisor through the edge nodes/ switches which enable links to be shared among multiple connections.

C. VALIDATION RESULTS

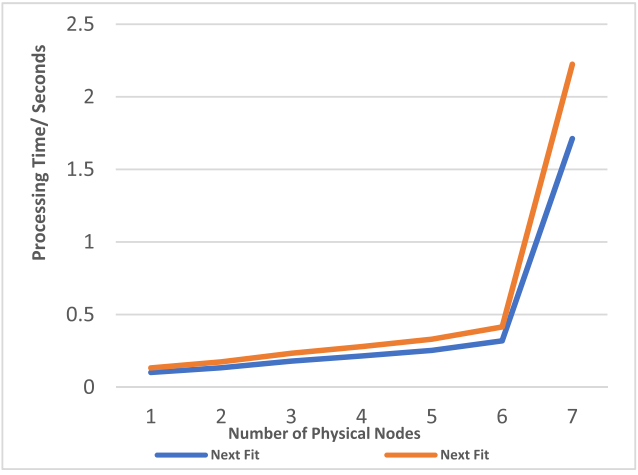
The diagram shown in Figure 6 displays three experimental scenarios which measure transmission time and processing time in a cloud network representing an automated 5G network deployment. Two of the diagrams evaluate placement policies in the cloud environments. In scenario 1 and scenario 2, variable data sizes were generated using the model proposed by [20] in Table 2 to measure processing network transmission time. The network latency of the links used in the datacentre was set to 1msec. Although variable data sizes and paths were used in these experiments, the difference in average data transmission time between CloudSimHypervisor and CloudSimSDN – NFV is about 9.3%. In scenario 1, transmission time was measured with the application of the First Fit policy in the cloud environment. Scenario 2 describes the rate at which data is transmitted between host networks and the Network Operating System (NOS) of CloudSimSDN–NFV and the Network Virtualization Hypervisor of CloudSimHypervisor. The difference in the transmission time observed in the two scenarios is due to delay based on the OSI layering of the two application. The end-to-end layering of the architecture of CloudSimSDN-NFV has more transmission layers, hence, has a slower rate to transmit



(a) Scenario 1



(b) Scenario 2



(c) Scenario 3

FIGURE 6. Measuring average processing speed and transmission speed with CloudSimHypervisor and CloudSimSDN-NFV.

network requests across the multiple independent network slices than CloudSimHypervisor.

The third scenario considers random distribution of virtual network requests to the host in the physical software defined

TABLE 2. Characteristics of network requests based on the distribution proposed by Ersoz *et al.* [20].

data	Distribution	parameters
Request inter-interval times	Log-normal Distribution	μ=1.5627, α=1.5458
Packet sizes	Log-normal Distribution	μ =5.6129, α =0.1343 (Ch1) μ =4.6455, α =0.8013 (Ch2) μ =3.6839, α =0.8261 (Ch3) μ =7.0104, α=0.8481 (Ch4)
Workload sizes	Pareto Distribution	Location = 12.3486, shape = 0.9713

TABLE 3. Link distribution for the validation experiments.

Link	Bandwidth
Hypervisor / Core ↔ Edge Switches	50 Mbps
Edge Switches ↔ Host Nodes	50 Mbps

TABLE 4. VM specification for joint compute and network resource utilization use case.

VM Type	Cores	MIPs	Bandwidth
Tenant controller	16	9000	500 Mbps
DB Server	12	2000	200 Mbps
App Server	8	2000	200 Mbps
Web Server	8	2600	200 Mbps
Intrusion Detection Server	8	3000	600 Mbps
Proxy Server	12	3500	600 Mbps

network based on the dynamic decisions of the Network Virtualization Hypervisor. This method is called the Hypervisor Fit Algorithm. In this scenario, data is randomly generated using the same model as the first two scenarios, however, fixed network paths were used as shown in Table 3.

The difference in average processing time between CloudSimHypervisor and CloudSimSDN-NFV is about 6.8% which is smaller compared to the first two scenarios. Table 4 displays the characteristics of the data which was used for the evaluating the accuracy of the CloudSimHypervisor with regards to network traffic prioritization.

Figure 7 displays results of the rate at which CloudSimSDN-NFV and CloudSimHypervisor execute workloads in an experiment in which fixed data size and fixed network paths were implemented as shown in Figure 5. The difference in executing VNE between CloudSimSDN-NFV and CloudSimHypervisor is about 2.5%. Furthermore, Figure 7 displays that CloudSimHypervisor executes more virtual network requests (close to 18.3%) given fixed data size and fixed network paths than CloudSimSDN-NFV.

This is because the differences in architecture of the two simulation frameworks affect factors such as fragmentation latency and delay across multiple infrastructure layers (Network Slices) which impacts network performance. For instance, whiles VNRs are immediately received by

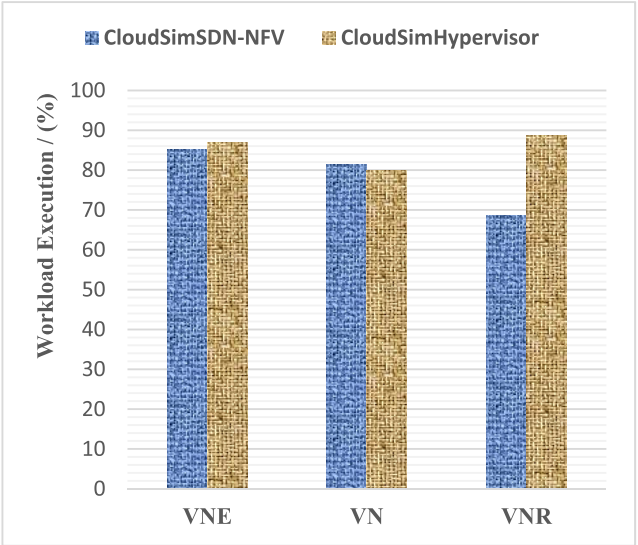


FIGURE 7. Workload Execution Rate in CloudSimHypervisor and CloudSimSDN-NFV with regards to average transmission speed.

the Network Virtualization Hypervisor in CloudSimHypervisor for processing, they are haphazardly placed in the CloudSimSDN-NFV framework and takes time to be group and processed. However, the results from the various experiments detailed in the above scenarios depict that the differences in accuracy between CloudSimHypervisor and CloudSimSDN-NFV simulation frameworks are based mainly on architectural differences.

VI. USE CASE EVALUATION

This paper focuses on two use cases (built in the context of multi-tier web applications) to demonstrate the capabilities of using CloudSimHypervisor and to highlight the strengths of adopting the Network Virtualization

Hypervisor in slicing networks in Software Defined Cloud Networks. The joint computer and network resource utilization is considered first. Then using the framework to further access how prioritizing network traffic affects network bandwidth usage and link resources considering different QoS parameters.

A. JOINT COMPUTE AND NETWORK RESOURCE UTILIZATION

The first use case evaluates the impact of using the network virtualization hypervisor to optimise usage of compute and network resources in a software defined cloud network. SDN enabled cloud datacentres enhance network resource utilization and energy savings via VM consolidation procedure [16]. Due to the consolidation of resources, switches and host which are unused can be turned off by the SDN controller. However, for software defined cloud networks (SDCN) which implements a network hypervisor, which supports several independent isolated virtual networks (Network Slices), enhances efficient use of these resources.

The Network Virtualization Hypervisor through the network specific tenant controllers accurately allocate network requests with specific requirements to the appropriate compute and network components which have adequate available resources to execute the requests. The Network Hypervisor has a global view of available resources in the entire network and the resources required for executing all pending task. The allocation of requests to available resources is done through allocation policies which are executed by the hypervisor with regards to certain performance parameters such as response time, transmission time and queuing time. Due to the efficiency with which the Network Virtualization Hypervisor allocates resources in SDCNs, there is efficient usage of hosts and switches resources with accurate measure of workload allocation. Hence, maximising energy conservation in SDCNs.

1) SETUP FOR EXPERIMENT 1

The first use case evaluates the impact of using the Network Virtualization Hypervisor on compute and network resources utilization in a software defined cloud network. In this scenario, we setup a large-scale cloud datacentre with 100 physical machines connected through 20 edge switches. The physical machines are configured with 16, 24 and 32 cores, and each core has 10,000 MIPS capacity. The network bandwidths of all links between switches and physical machines were equally set to 2 Gbps. The Network Virtualization Hypervisor is setup to receive workloads from network user requests which consist of CPU processing and network transmission requirements and then pass it to the physical networks for execution through the edge switches.

For the virtual topology, 500 VM creation requests of which 100 are tenant controllers were randomly generated based on selected VM types specified in Table 2. The tenant controllers in the simulation are configured to mediate between the set of VMs forming a particular virtual network and the hypervisor. Each of these requests has a different start time and lifetime following exponential distribution and Pareto distribution respectively [20] which we adopted from [7]. To ensure that switches are working throughout the VM lifetime, network workload was also created for the execution time of VMs [16].

Efficiency of the Network Virtualization Hypervisor in maximizing resource utilization in software defined cloud networks (SDCN) is assessed in a series of experiments using the CloudSimHypervisor simulation framework. The diagrams in Figure 8 display two experimental scenarios measuring different network performance metrics using fixed number of hosts of 100 nodes for traditional SDN architectural setups where the network control is the SDN controller and for SDCN architectural setups where the network hypervisor is the network control. Figure 8a., shows that given the same number of network host resources, software defined cloud networks (SDCN) which implements the Network Virtualization Hypervisor optimized bandwidth usage by close to 21.43% and CPU usage by close to 67.31% compared to

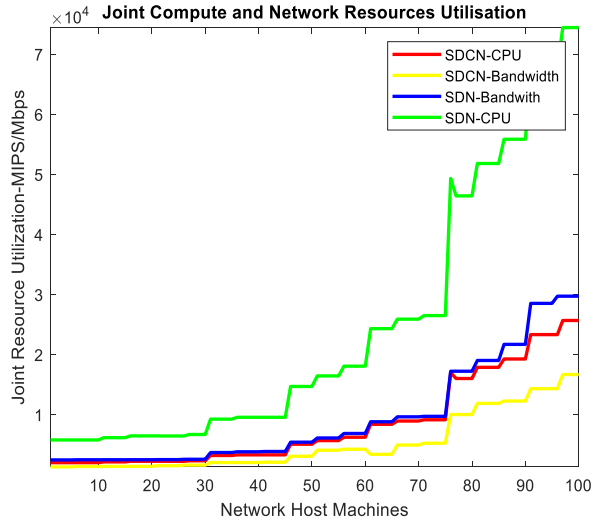


FIGURE 8. Efficient utilization of compute, network resources and energy consumption with using the network virtualization hypervisor.

traditional SDN architecture which is managed by the SDN controller. Figure 8b. also depicts that the Network Virtualization Hypervisor conserved energy during the experiment by close to 23.33% compared to the energy consumed by the SDN controller for the same experiment. These results explain that software defined cloud networks which implement the Network Virtualization Hypervisor efficiently optimize usage of network resources and are more energy efficient as compared to the traditional software defined networks.

B. TRAFFIC PRIORITIZATION

Prioritizing network traffic based on the user type was difficult due to complexity and overhead of configuring network elements in traditional cloud datacentre networks. Software defined networking provided a dynamic cloud environment which enhanced this challenge through network

traffic consolidation and VM placement techniques [16]. However, the emergence of network virtualization in software defined cloud networking environments and the use of the Network Virtualization Hypervisor which enables multiple independent virtual networks to co-exist and share the same physical network infrastructure introduce a dynamic priority-aware network request and traffic micro-segmentation and bandwidth allocation for different network user types with efficient QoS delivery.

Each network user request consists of a random number of VMs and flows with thorough specification. Priority of a request is accessed from the specification of the VM and flows by the Network Virtualization Hypervisor. Based on the analysed information, the network hypervisor dynamically implements the appropriate host and link selection algorithm for bandwidth allocation and flow scheduling to execute requests for all network user types.

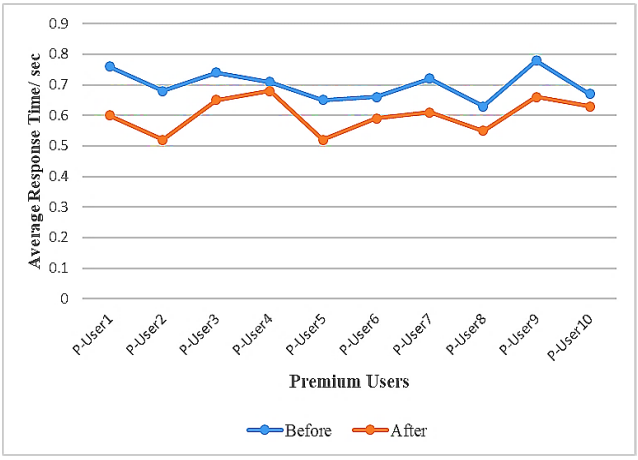
1) SETUP FOR EXPERIMENT 2

In this scenario, we simulate a cloud datacentre network with 100 physical machines connected through 10 edge switches. The physical machines are configured with 16, 24 and 32 cores, and each core has 20,000 MIPS capacity. The network bandwidths of the links of the host-network were randomly allocated at 1 Gbps, 2 Gbps and 4 Gbps with each having a 0.5 msec latency. In the simulation environment, the network virtualization hypervisor is able to create different channels for data flows in order to provide priority network traffic with the additional bandwidth demand. Which implies virtual channels (links) connecting the VMs are dynamically set to differentiate higher priority flows over normal flows for all network user types; premium, standard and ordinary. Standard channels by default evenly distribute all packets and transfer data in the channels when there's no network traffic prioritization. However, the network hypervisor allocates a specific amount of bandwidth exclusively for the priority channel, when traffic prioritization is required. Hence, the bandwidth in these channels is not available for the other channels.

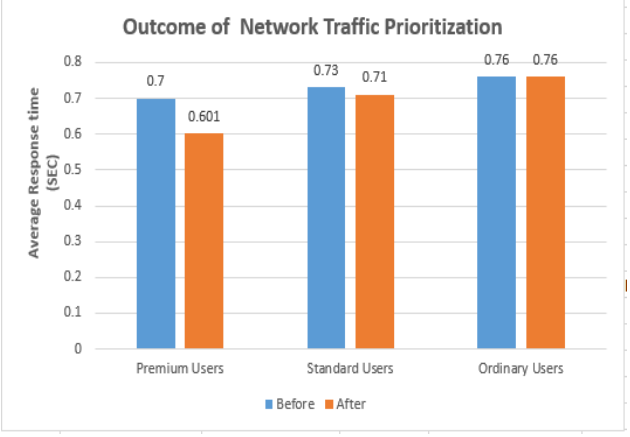
Different workloads are generated for each user using the web model [20]. Each of these requests has a different start time and lifetime following exponential distribution and Pareto distribution respectively. 30 cloud infrastructure customers for the experiment have been used, 10 of them were premium users, 15 standard, and 5 ordinary users.

Figure 9a displays details of performance improvement for premium users with the implementation of network traffic prioritization. The average response times for network request for all Premium Users received an improvement of about 23.75%. This also implies that the network virtualization hypervisor is efficient at dynamically allocating network bandwidth per- flow to users, an attribute which is important in enhancing QoS in software defined cloud networks.

Figure 9b, shows that average response time of Premium User requests decreased from 0.7 seconds to 0.61 seconds which is an average of about 14.143% performance



(a) Average response time premum user



(b) Traffic outcome

FIGURE 9. Effect of network traffic prioritization.

enhancement. The figure also shows an improvement of approximately 2.74% in the response time for Standard User. However, the response time for Ordinary User, remained the same. This explains that with the implementation of the network virtualization hypervisor in software defined cloud networks, Cloud Service Providers would be able to efficiently deliver services to users with different QoS requirements.

VII. CONCLUSION

Advances in technology and the explosion of digital content, cloud computing, 5G and 6G Technologies have introduced many innovative technologies and a paradigm shift in network infrastructure deployment. Modern networking and cloud technologies such as SDN, NFV and Edge Compute have attracted attention from industry and academia. Several State- of- the-Art have been proposed in literature and some developed testbeds. However, the cost involved in developing in these testbeds and the limitations with the existing simulation applications demands a simulation framework which has functions that addresses these challenges and capabilities which advances further studies

and applications of theory to development of new technologies. In this paper the CloudSimHypervisor is presented demonstrating an innovative simulation framework to model and evaluate Network Slicing and end-to-end heterogenous network automation which are key enablers of 5G and 6G networking technologies. CloudSimHypervisor was developed by extending CloudSimSDN-NFV framework which is an extension of the well-studied and used CloudSimSDN and CloudSim toolkits. Two use case experiments were conducted with this simulation framework in SDN, NFV and Edge compute environment to validate its accuracy. Results from these experiments display the efficiency of the CloudSimHypervisor in evaluating processing speed, transmission speed, compute and network usage efficiency and energy consumption.

VIII. FUTURE WORK

The proposed simulation framework can be improved by implementing features and strategies in artificial intelligent features such as Artificial Neural Networks (ANN) and related enabling technologies to expand the scope of target groups in the research community and in industry. 5G and 6G networking technologies comprise peripherals from multiple vendors with different specifications. Features and resources to simulate these components would be introduced in future studies. Due to the inconsistent nature of network traffic at different periods of the day, resource utilization algorithms such as overbooking should not be set to have static ratios for multiple heterogeneous networks. Provision would also be made for workloads for network user requests with different QoS requirements and parameters. The objective is to make CloudSimHypervisor a viable tool for the research community to aid in modelling and simulating innovative network architectures for 5G and 6G networks.

REFERENCES

[1] J. Son, T. Z. He, and R. Buyya, "CloudSimSDN-NFV: Modeling and simulation of network function virtualization and service function chaining in edge computing environments," *Softw., Pract. Exper.*, vol. 49, no. 12, pp. 1748–1764, 2019.

[2] D. Kreutz, F. M. V. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-defined networking: A comprehensive survey," *Proc. IEEE*, vol. 103, no. 1, pp. 14–76, Jan. 2015.

[3] S. K. Garg and R. Buyya, "NetworkCloudSim: Modelling parallel applications in cloud simulations," in *Proc. 4th IEEE Int. Conf. Utility Cloud Comput.*, Melbourne, VIC, USA, Dec. 2011, pp. 105–113, doi: 10.1109/UCC.2011.24.

[4] R. Buyya, R. N. Calheiros, J. Son, A. V. Dastjerdi, and Y. Yoon, "Software-defined cloud computing: Architectural elements and open challenges," in *Proc. Int. Conf. Adv. Comput., Commun. Informat. (ICACCI)*, Sep. 2014, doi: 10.1109/ICACCI.2014.6968661.

[5] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer, "Survey on network virtualization hypervisors for software defined networking," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 655–685, 1st Quart., 2016, doi: 10.1109/COMST.2015.2489183.

[6] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network function virtualization: State-of-the-art and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 236–262, 1st Quart., 2016, doi: 10.1109/COMST.2015.2477041.

[7] A. Blenk, A. Basta, and W. Kellerer, "HyperFlex: An SDN virtualization architecture with flexible hypervisor function allocation," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage. (IM)*, May 2015, pp. 397–405, doi: 10.1109/INM.2015.7140316.

[8] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw., Pract. Exper.*, vol. 41, no. 1, pp. 23–50, Jan. 2011, doi: 10.1002/spe.995.

[9] I. F. Akyildiz, S. Nie, S.-C. Lin, and M. Chandrasekaran, "5G roadmap: 10 key enabling technologies," *Comput. Netw.*, vol. 106, pp. 17–48, Sep. 2016.

[10] M. Ruffini, "Multidimensional convergence in future 5G networks," *J. Lightw. Technol.*, vol. 35, no. 3, pp. 535–549, Feb. 1, 2017.

[11] C-I. Fan, Y.-T. Shih, J.-J. Huang, and W.-R. Chiu, "Cross-network-slice authentication scheme for the 5^{th} generation mobile communication system," *IEEE Trans. Netw. Service Manage.*, vol. 18, no. 1, pp.701-712, Jan. 2021.

[12] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, May 2017.

[13] B. Lantz, B. Heller, and N. McKeown, "A network in a laptop: Rapid prototyping for software-defined networks," in *Proc. 9th ACM SIGCOMM Workshop Hot Topics Netw. (Hotnets)*, 2010, p. 19.

[14] A. Núñez, J. L. Vázquez-Poletti, A. C. Caminero, G. G. Castañé, J. Carretero, and I. M. Llorente, "ICanCloud: A flexible and scalable cloud infrastructure simulator," *J. Grid Comput.*, vol. 10, no. 1, pp. 185–209, Mar. 2012, doi: 10.1007/s10723-012-9208-5.

[15] M. S. Aslanpour, A. N. Toosi, J. Taheri, and R. Gaire, "AutoScaleSim: A simulation toolkit for auto-scaling Web applications in clouds," *Simul. Model. Pract. Theory*, vol. 108, Apr. 2021, Art. no. 102245.

[16] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, "CloudSimSDN: Modeling and simulation of software-defined cloud data centers," in *Proc. 15th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput.*, May 2015, pp. 475–484.

[17] S. Chaabnia and A. Meddeb, "Slicing aware QoS/QoE in software defined smart home network," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp. (NOMS)*, Taipei, Taiwan, Apr. 2018, pp. 1–5.

[18] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in *Proc. 10th IEEE/ACM Int. Conf. Cluster, Cloud Grid Comput.*, May 2010, doi: 10.1109/CCGRID.2010.46.

[19] J. Son and R. Buyya, "A taxonomy of software-defined networking (SDN)-enabled cloud computing," *ACM Comput. Surveys*, vol. 51, no. 3, pp. 1–36, Jul. 2018, doi: 10.1145/3190617.

[20] D. Ersoz, M. S. Yousif, and C. R. Das, "Characterizing network traffic in a cluster-based, multi-tier data center," in *Proc. 27th Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2007, p. 59.

ANDREWS O. NYANTEH received the bachelor's degree in computer science and physics from the University of Cape Coast, Ghana, the Master of Science degree in information technology from the Sikkim Manipal University of Health, Medical and Technological Sciences, India, and the Ph.D. degree in computer engineering from Brunel University London, in 2020. His research interests include software defined networks, computer and mobile networking and cloud unification, network resources and functions virtualization, edge computing, 5G and 6G technologies, and the Industrial Internet of Things.

MAOZHEN LI received the Ph.D. degree from the Institute of Software, Chinese Academy of Sciences, in 1997. He is currently a Professor with the Department of Electronic and Computer Engineering, Brunel University London, U.K. His main research interests include high performance computing, big data analytics, and intelligent systems with applications to smart grid, smart manufacturing, and smart cities. He has over 160 research publications in these areas, including four books. He is a fellow of the British Computer Society and IET. He has served over 30 IEEE conferences and on the editorial board of a number of journals.

MAYSAM F. ABBOD (Senior Member, IEEE) received the Ph.D. degree in control engineering from The University of Sheffield, U.K., in 1992. He is currently a Reader in intelligent systems with the Department of Electronic and Computer Engineering, Brunel University London, U.K. He is also a chartered engineer in U.K. He has authored more than 50 articles in journals, nine chapters in edited books, and more than 50 papers in refereed conferences. His current research interest includes intelligent systems for modeling and optimization. He is a member of IET, U.K. He is also serving as an Associate Editor for the *Engineering Applications of Artificial Intelligence* (Elsevier), *Sensors* (MDPI), and *Electronics* (MDPI).

HAMED AL-RAWESHIDY (Senior Member, IEEE) received the Ph.D. degree from the University of Strathclyde, Glasgow, U.K., in 1991. He was with the Space and Astronomy Research Centre, Iraq; PerkinElmer, USA; Carl Zeiss, Germany; British Telecom, U.K.; the University of Oxford, U.K.; Manchester Metropolitan University, U.K.; and the University of Kent, U.K. He is currently the Director of the Wireless Networks and Communications Centre (WNCC) and PG studies (ECE) with Brunel University London, U.K. WNCC is the largest centre at Brunel University and one of the largest communication research centre in U.K. He is also a Professor of communications engineering with the University of Strathclyde. He is also an Editor of the first book in *Radio over Fibre Technologies for Mobile Communications Networks*. He acts as a consultant and involved in projects with several companies and operators, such as Vodafone, U.K.; Ericsson, Sweden; Andrew, USA; NEC, Japan; Nokia, Finland; Siemens, Germany; Franc Telecom, France; Thales, U.K.; Thales, France; and Tekmar, Italy. He is also a principal investigator for several EPSRC projects and European project, such as MAGNET EU Project (IP), from 2004 to 2008. He has published more than 400 journal articles and conference papers. His current research interests include beyond 5G and 6G, such as C-RAN, SDN, the IoT, MMwave 240-400 GHz, UMMIMO, M2M, and QKD, and Radio over Fibre.