Models and algorithm of optimization launch and deployment of virtual network functions in the virtual data center

I P Bolodurina and D I Parfenov

1 Department of Applied Mathematics, Orenburg State University, Pobedy avenue, 13, Orenburg, Russia
2 Faculty of Distance Learning Technologies, Orenburg State University, Pobedy avenue, 13, Orenburg, Russia

e-mail: parfenovdi@mail.ru

Abstract. The goal of our investigation is optimization of network work in virtual data center. The advantage of modern infrastructure virtualization lies in the possibility to use software-defined networks. However, the existing optimization of algorithmic solutions does not take into account specific features working with multiple classes of virtual network functions. The current paper describes models characterizing the basic structures of object of virtual data center. They including: a level distribution model of software-defined infrastructure virtual data center, a generalized model of a virtual network function, a neural network model of the identification of virtual network functions. We also developed an efficient algorithm for the optimization technology of containerization of virtual network functions in virtual data center. We propose an efficient algorithm for placing virtual network functions. In our investigation we also generalize the well renowned heuristic and deterministic algorithms of Karmakar-Karp.

1. Introduction

The technology of cloud computing is built on the basis of the virtualization of the individual components that make up the data center infrastructure. Approaches to the organization of the virtualization layer are divided according to the levels of application of this technology. Typically, the following types of virtualization are distinguished: operating system, software, memory, data storage, databases and network. The most active development level is the virtualization of the network. It allows you to create a virtual environment to run multiple instances of user space within the same system used to run network service and functions within the cloud environment of virtual data center.

Nowadays virtual data center is representing as are a fairly complex multi-level mechanism that interacts in the course of its work with various objects and data flows of the network infrastructure [1, 2]. To analyze more data, it is necessary to apply the processing using methods used in Big Data.

One of the approaches to virtualization, which allows implementing this approach, is containerization technology. A container is an object that provides a user with access to necessary libraries and contains the required set of software for launching the development environment, a ready application or a service. Furthermore, the advantage of using virtualization on the basis of containers for placing applications and services in a network environment of a virtual data center is the lightness of the solution, which allows to reduce a number of costs and to increase the productivity of cloud-
based service-oriented applications. As a containerization technology, the most effective is the control system based on Docker [3]. The advantage of this technology is its reliability, as well as the availability of open source code and a convenient API for sharing in a networked virtual data center environment. A significant drawback of containers is the inability to provide the proper level of isolation of data flows, as well as the lack of support for migration between compute nodes in real time.

A key difference between virtual machine virtualization and virtualization is the use of a hypervisor that emulates the hardware of a physical computing node. Within each virtual machine, a full-fledged operating system functions. All this leads to significant overhead in terms of resource consumption in a virtual data center.

As a result of investigations, we developed a solution that allows solving the problem of container migration, as well as contributing to optimization of overhead costs for computational resources for virtual data center infrastructures. The developed solution is based on hybrid virtualization, which is a combination of two approaches to the deployment virtual network functions in a software-managed infrastructure of a virtual data center: a container and based on virtual machines.

With resources virtualization technology development, the number of layers is steadily increasing, forming infrastructure decisions and being used in cloud computing technology. Nowadays, there can be defined up to six levels of software-defined infrastructure of virtual data centers, which use modern cloud platforms for deployment.

2. The level distribution model of software-defined infrastructure virtual data center

Let us introduce the level distribution model of software-defined infrastructure of virtual data center, which support the containerization method of network application and services disposal in cloud systems. The first level – the hardware component of any data center, which includes computational nodes (Nodes), file systems (Storages) and physical network units (NetObj). Let us introduce it as the set of decisions: \( \text{PhysLayer} = \{\text{Nodes}, \text{Storages}, \text{NetObj}\} \).

The next level represents the software-defined layer. This layer consists of the same number of objects as the first level but the main difference is that all the infrastructure elements are dynamic, easily-transformed and adjusted within the limits of the physical database network environment. The second level can be presented as the next set of connections:

\[
\text{SDLayer} = \{\text{SDNodes}, \text{SDStorages}, \text{SDNetwork}\},
\]

where are: \( \text{SDNodes} \) – software-defined computing units; \( \text{SDStorages} \) – software-defined storages; \( \text{SDNetwork} \) – software-defined network.

Above the layer of software-defined infrastructure, there is a level of the specific objects virtualization. Most of them are: computing nodes (VirtNodes), virtual data warehouses and also the elements of software-defined networks that are used in works of cloud platforms and consolidated in VirtNetwork multitude.

\[
\text{VirtLayer} = \{\text{VirtNodes}, \text{VirtStorages}, \text{VirtNetwork}\},
\]

In the software-defined infrastructure, computing nodes and data storages are more often presented as virtual machines that discharge the set of given functions.

To control such multi-layer infrastructure, the separate orchestration layer is needed (forth level). There are a number of functions in it and computing nodes and software systems are enabled for their execution. The main functions are orchestration of the virtualization objects (virtual machines and data storages) (ONodes, OStorages) and orchestration of the software-defined network (ONetwork), which include network function virtualization.

\[
\text{OrchLayer} = \{\text{ONodes}, \text{OStorages}, \text{ONetwork}\}
\]
The next level (Service level) represents the services used in the working process either the very cloud platform, or applications distributed there, for example, DBMS, Hadoop, Nginx and others.

\[
\text{ServiceLayer} = \{\text{Service}_1, \ldots, \text{Service}_n\} \tag{4}
\]

All the multitude of ServiceLayer network services working in the virtual data center infrastructure can be divided into two disjoint subsets \( \text{ServVM} \cup \text{ServDocker} = \text{ServiceLayer} \). Services using virtualization on the base of other machines are in the first set (ServVM). In the second set (ServDocker), there are services realized on the base of containers under Docker control.

On the top level, there are network applications that are exploited by users for flexible scalability providing (AppLayer).

\[
\text{AppLayer} = \{\text{App}_1, \ldots, \text{App}_m\} \tag{5}
\]

As on the previous level, network applications \( \text{App}_i \) can be placed in containers, forming the AppVM set or use containerization forming the AppDocker set. At the same time \( \text{AppVM} \cup \text{AppDocker} = \text{AppLayer} \).

Thus, the set of object of software-defined infrastructure can be divided into two groups by the methods of placing. Virtual objects using container placing method can be referred to the first group. Let us describe them in the following way:

\[
\text{Docker} = \{\text{ServDocker}, \text{AppDocker}\} \tag{6}
\]

In the second group there are services and applications using virtual machines as a placing platform:

\[
\text{VM} = \{\text{ServVM}, \text{AppVM}\} \tag{7}
\]

Before we talk about the ways of placing network applications and services, which realized virtual network functions in a virtual data center, we need to determine their structure, basic parameters, and key characteristics of their operation affecting the efficiency of their use. For this purpose, we have developed a generalized model of a virtual network function.

3. The generalized model of a virtual network function

The specific feature of virtual network function is the approach, where users have access to them and to their services; however, they do not know anything about their actual location. In most cases, users only know the address of the aggregation node and the application name. The orchestrator of virtual data center automatically selects the optimal virtual machine for the request, on which it is to be processed.

The generalized model of a virtual network function is a multilayer structure, described in a form of graphs to characterize the connections of individual elements. The model can be represented in the form of three basic layers, detailing the connections of the specific objects of virtual data center infrastructure: network applications, network services and network resources.

The network applications are a weighted directed acyclic graph of data dependencies:

\[
\text{NFVAappl} = (G, V) \tag{8}
\]

Its vertices \( G \) are tasks that get information from the sources and process it in accordance with the user requests; its directed edges \( V \) between corresponding vertices are a link between tasks in a schedule plan. The schedule plan is defined as a procedure which is prepared to follow the user's request (SchemeTask). Each vertex \( g \in G \) is characterized by the following tuple:

\[
\text{NFVAappl} = (\text{Res}, \text{NAppl}, \text{Uttime}, \text{SchemTask}) \tag{9}
\]
where are $Res$ are the resource requirements; $NAppi$ is the number of application instances; $Utime$ is the estimated time for of the users’ request execution; $SchemeTask$ is a communication scheme of data transmission between sources and computing nodes.

Each directed edge $v \in V$ connects the network application with the required data source. It is characterized by the following tuple:

$$v=(u, v, Tdata, Mdata, Vdata, Qdata).$$

where $u$ and $v$ are linked vertices; $Tdata$ is the type of transmitted data; $Mdata$ is the access method to the data source; $Vdata$ is the traffic volume estimated by the accessed data (in Mb), $Qdata$ is the requirements for the QoS (quality of services).

The model is original, because it enables to calculate the consolidated assessment of its work with data sources for each network application. It allows to predict the performance of the network in whole virtual data center.

As mentioned above, a network service is one of the key slices within the generalized model of a virtual network function. The network service is an autonomous application. Generally, the network service is highly specialized and designed to perform a limited set of network functions. The advantage of using the network service is isolated data flow processing. The usage of network services as virtual network functions reduces the execution time for user requests. The network service is described as a directed graph of data dependencies. The difference lies in the fact that from the user’s point of view, the network service is a closed system. The network service can be formalized as a tuple:

$$NetServ=( AğrIP, NameServ).$$

where are $AğrIP$ is the address of aggregation computing node; $NameServ$ is the network service name.

The aggregator of a network service selects an optimal virtual machine; it is executed on this machine. In addition, all its applications are distributed between predefined virtual machines or physical servers. In addition, all its network applications are distributed between predefined virtual machines or physical servers. Their new instances are scaled dynamically depending on the volume of incoming traffic.

In order to describe the placement of network applications and network services in the virtual data center infrastructure, we have also implemented the model of a network resource. A network resource is an object of a virtual data center, which describes the behavior and characteristics of the virtual network function, depending on its current state and parameters. The objects of virtual data center are disk arrays including detached storage devices, virtual machines, software-defined storages. In addition, each network service or network application imposes requirements on the number of computing cores, RAM and disk sizes, and the presence of special libraries on physical or virtual nodes used to launch their executing environments. Each network resource can be formalized as follows tuple:

$$NetRes=(TRes, Param, State, Core, Rmem, Hmem, Lib),$$

where are $TRes$ is the type of resource; $Param$ is the set of parameters; $State$ is the state of resource; $Core$ is the number of computing cores; $Rmem$ is the size of RAM; $Hmem$ is the size of a disk; $Lib$ are libraries requirements.

The distinctive feature of the model suggested implies analyzing network resources from the user viewpoint and from the viewpoint of a software-defined infrastructure of the virtual data center. The model is innovative, since it simultaneously describes the virtual network function placements and the state of the virtual environment, taking into account the network topology.
4. The neural network model of the identification of virtual network functions in the infrastructure of the virtual data centers

We have chosen Kokhonen’s network as a neural structure for modeling, since it is the most efficient in the clustering and classification of objects. Another important factor is the visualization of results; it enables to improve the understanding of the structure and character of data at early stages and to specify a neural network model further. Due to the peculiarities of network function virtualization, the support of classification in Kokhonen's network can be used to identify the uniform elements in network for further optimization of their placement. Kokhonen's network is trained by a method of consecutive approximations. Starting from the initial placement of objects selected randomly, the algorithm gradually improves it to supply the data clustering. Another advantage of Kokhonen's network is the opportunity to identify new clusters. The trained network detects clusters in the training data and refers all the data to certain clusters. If the network meets a set of data, which differ from any known samples, it will independently reveal a new cluster of elements then. This feature is very relevant, since it allows entering new network functions into the architecture of virtual data center without the actual change of algorithms of their distribution on physical and virtual computing nodes.

The principle of creating a neural network system to optimize the placement of network function virtualization in the multi cloud environment of the virtual data center is as follows. We have selected a number of criteria using the data obtained from the systems of virtual data center monitoring. This enables to both identify the virtual network function and assess its load on computing nodes. Criteria are formulated so that the answer could be always represented in the binary form, i.e. 1 is "Yes" or 0 is "No". The obtained data enable to form the vector of signals \( E = \{ e_1, e_2, \ldots, e_n \} \), which is placed at the entrance of a neural network. The vector of output values is similar, it has binary components.

The neural network is a two-dimensional matrix of neurons of dimension \( n \) (the number of inputs of each neuron) per \( m \) (the number of neurons). The number of inputs of each neuron is determined with respect to the number of criteria established earlier. The amount of neurons \( m \) coincides with the required number of classes and corresponds to the number of the unique network functions used in work of a multi cloud platform. The importance of each of the entrances to neuron is characterized by the numerical size called by weight. It is set in the form of matrix:

\[
X = \begin{pmatrix}
 x_{11} & x_{12} & \ldots & x_{1j} \\
 x_{21} & x_{22} & \ldots & x_{2j} \\
 \vdots & \vdots & \ddots & \vdots \\
 x_{i1} & x_{i2} & \ldots & x_{ij}
\end{pmatrix}
\]

(13)

With the vectors of weight coefficients of connections \( x_{ij} = \{ x_{ij,1}, x_{ij,2}, \ldots, x_{ij,n} \} \) as its elements.

Kokhonen’s network consists of three layers of neurons. The basis of the network is a covert Kokhonen’s layer. However, in this research, we have offered a changed scheme of output neurons of Kokhonen’s network to obtain the results to identify destination and, simultaneously, to find critical loading on a calculating node.

We offered to divide the covert layer of Kokhonen’s neural network into two sets. The first set of neurons \([1…K]\) is responsible for the identification of a network function placed in the virtual data center. The work of neural network changes input scales at the exit layer activates the linear function \( Y_1 \), which takes the value \([0…n]\), where 0 means that the network object under study has no signs of a virtual network function, and values from 1 to \( n \) correspond to a particular network function identified by a neural network model. The second set of neurons \([L…Z]\) analyzes the loading of the network object under study and initiates the function \( Y_2 \) at the exit, which take the values \([-1,0,1]\), where 0 is a normal state, -1 means that the network function is idle or does not perform its functions, and 1 means that the network function is overloaded.
The basic criteria used as input data to detect virtual network functions are network records and events, data of the time of packages going through a network object, time of packet input and output, memory loading, the use of CPU, the intensity of dataflow, TTL and others. The data collected in the network are placed at the entrance of a neural network and create a full neural network. That being the case, we should simplify and verbalize a neural network by excluding some elements without the significant reduction of the detection quality.

To train a neural network, we have used the data obtained from the system of monitoring of virtual data center of the Orenburg State University. It includes 4 OpenFlow switches (2 x HP 3500yl, 2 x Netgear GSM7200), 8 computing nodes (32Gb RAM, 4 cores), 1 server (32Gb RAM, 8 cores) with OpenFlow controller and 1 server (32Gb RAM, 4 cores) for monitoring function. Routers connected compounds having the speed 1000 Mbit / s, and the computers are connected to a third level router via the second level network connections at 100 Mbit/s. We have chosen the most popular virtual network functions for the experimental study (Router, NAT, Firewall, Proxy, Switch, DPI).

Table 1. The result of experimental identification of virtual network functions

| Virtual network function | The number of instances | Correct detection |
|--------------------------|-------------------------|-----------------|
| vRouter                  | 20                      | 19 (98%)        |
| vNAT                     | 15                      | 13 (94%)        |
| vFirewall                | 18                      | 17 (93%)        |
| vProxy                   | 25                      | 24 (96%)        |
| vSwitch                  | 30                      | 28 (93%)        |
| vDPI                     | 16                      | 14 (87.5%)      |

The obtained data enable to conclude that the developed neural system had some difficulties with the detection of some network functions because of their small differences in the parameters chosen. This defect can be eliminated by the introduction of additional criteria to the initial model of neural network. Thus, the application of the developed neural network system enables to identify virtual network functions correctly in middle in 94% of cases, while the resource intensity is insignificant. This increases the efficiency of optimization for launch them in the infrastructure of a multi cloud platform.

5. The algorithm of optimization launch and deployment of virtual network functions in the virtual data center

The presented models allow us to choose the most suitable methods of launch and deployment instance of virtual network functions in virtual data center infrastructures based on current load and the incoming flow of requests. The main task of the distribution of virtual network functions is to choose the number of instances in time interval, which formulate as making a plan. When accessing the virtual network functions, the preparation of a plan is especially important. Due to the load, the compute nodes may vary widely over relatively short time intervals, and depends on the method of placement of the virtual data center infrastructure. To solve the optimization problem, an algorithm was developed, which monitors the virtual data center infrastructure, scheduling and launching applications and services. It is based on a biased random-key genetic algorithm (BRKGA). But different to BRKGA, the algorithm uses heuristic analysis queries flows and their classification depending on the application hosted in the virtual data center.

Our model of identification of virtual network functions allows optimizing their launch and deployment in the virtual data center. We will optimize the launch and deployment of the network functions found in the virtual data center by using Kokhonen's network by the following criteria: the current load created on computing nodes; resource intensity of a network function; the number of flows going through computing nodes. The main objective of virtual network functions launch and deployment is to choose the optimum number of the nodes to implement required functionality
as a software solution. Thus, there is a problem of resource planning. Planning is of particular relevance in the organization of dynamic topology in the virtual data center, since the load on computing nodes can change over a wide range at rather short intervals of time and depends on the chosen type of specific network functions placement. To solve the task of optimization, we have developed the algorithm for monitoring the infrastructure of the virtual data center as well as for the placement and launch of network functions. In comparison with the available analogs, the algorithm uses the heuristic analysis of traffic flows and their classification depending on the type of a network function.

The general algorithm has the following sequence of steps.

Step 1. To identify the arrangement of virtual network functions taking into account the topology of the network infrastructure of the virtual data center.

Step 2. To estimate the number of the launched copies of each virtual network function and to range them according to the popularity of the network infrastructure. The popularity is estimated towards traffic flows, which run through the launched copies of a virtual network function.

Step 3. To define the load on physical and virtual computing nodes created by each copy of function.

Step 4. To compare data and to define virtual network functions, which demand scaling or folding, using the data obtained in steps 1 and 2.

Step 5. To reconfigure the topology on the controller of the software-defined network, stop and release the occupied resources of virtual data centers for network functions, which require folding.

Step 6. To evaluate a method of placement for virtual network functions, which require scaling and creation of the maximum load on the infrastructure. To distribute the most loaded network function using a hybrid way of placement (the containers developed in the virtual machine). To transfer network functions, which are less loaded but require scaling, to the operating mode of a virtual machine.

Step 7. To provide the migration of the virtual machines with network functions on the least loaded computing nodes.

The approach in this algorithm for controlling the launch and deployment of virtual network functions enables to take into account a method of placement and to organize the work of the virtual data center with account for circulating traffic flows and regulate the number of the launched copies of each function.

6. Experimental part

The aim of the experimental research is to define the effectiveness of the use algorithm of placement the virtual network functions in infrastructure of the virtual data center. For each type of virtual network functions created images templates of container and virtual machine for deployment in infrastructure of the virtual data center. On the basis of statistical data using queries created by the flows generator are simulated requests from users to cloud applications. Flows of different intensity were used to evaluate the performance. In the first case, flows create a minimum load capacity (to evaluate response times and delays, which makes data center infrastructure) (exp 1). In the second case, workload was created. We generated the traditionally values of traffic for each cloud application (exp 2). In the third case, a developed algorithm of optimization launch and deployment of virtual network functions (exp 3).
The investigation has shown that using static placement of virtual network functions in the containers on physical nodes is not effective, because it does not allow redistributing the load quickly. In addition, the movement of the container to another computing unit leads to a loss of current connections. Placing the virtual network functions based on virtual machines due to the flexibility of load balancing showed better results. However, loading on compute nodes has increased considerably due to the additional overhead associated with the use of virtual machines. The most effective placement of the study was the use of containers inside the virtual machines. This solution was possible to increase the density of placement virtual network functions within virtual data centers, as well as to allow placing containers, data services and network applications in close proximity to each other, thereby reducing the response time of applications on users’ queries and thus increasing the efficiency of the cloud system.

7. Discussion
The variety of physical network devices of various vendors increases both capital expenditure and operational costs for the maintenance of data centers. The technology of network function virtualization allows solving this problem by the realization of network functions as software. The application of the NFV implies the use of the technology of network objects virtualization, which function as software and particular computing processes, or as complex infrastructures of cloud computing instead of hardware solutions.

The group of scientists headed by Liqiong Chen [4] analyzes the architecture and mechanisms of the interaction between the technology of network function virtualization and the software-defined networks. As noted in the research, if the number of the users who launch applications in the multi cloud network environment increases, there is a competition for resources. Besides, each user request is described by the relevant requirements to network environment from the viewpoint of productivity, safety, and the effective control of objects in virtual infrastructure.

Scientists from Arizona State University [5] have studied a multi cloud system. They offer an approach to the creation of network architecture based on the NFV technology as alternatives for traditional hardware network devices. However, this research does not solve the main problem concerning the methods of NFV placement on computing nodes.

Apart from a problem of the effective placement of network functions, the NFV technology has a number of disadvantages associated with the organization of the coordinated control of the entire network infrastructure of a virtual data center. To solve this problem, scientists from University of Wisconsin-Madison have proposed an approach, which is a framework of the OpenNF [6]. This approach provides the effective coordinated control of both the internal state of network functions and the state of data transmission network. However, this decision does not solve another important issue associated with the overall performance and load on the controller and objects of network functions.

Figure 1. Computational experiment result
The scientists from Nation Chung Cheng University have investigated the reduction of load on the controller to ensure the work of the NFV technology [7]. As a rule, in case of the software-defined network use, the controller classifies the traffic received from the ports on network nodes to define a path to network objects, which play a role of the virtual network functions. This process generates large volumes of traffic in the plane of control. The authors offer the expansion for the architecture based on the software-defined network to reduce the load created by network traffic on the controller due to the use of the NFV technology. The solution represents two-layer classification of traffic based on the OpenFlow protocol. Network events are analyzed in the plane of data instead of the plane of control.

The group of scientists headed by Hassan Jameel Asghar has developed a scalable system to work with the technology of network function virtualization in the multi cloud environment [8]. The authors have offered an abstract model of the standard network function distributed between several cloud platforms. The developed model is used as a basis for the SplitBox system. This system enables to increase the speed of package processing considerably in comparison with similar hardware solutions. An essential disadvantage of the concept is the resource intensity of this system. The overhead costs of the deployment and work of the Splitbox network functions demand the same quantity of computing resources as in case of hardware network devices. This disadvantage neutralizes the available advantages of the approach, because the problem of the effective placement of network functions in the multi cloud infrastructure of the virtual data center remains unsolved.

Thus, it is established that the technology of network function virtualization and the existing cloud solutions on based on the software-defined infrastructure of the virtual data center has a number of advantages, which enable to improve the quality of service in data transmission networks. However, today, there are adequate and effective solutions, that would enable to control the placement of the virtual network functions on physical and virtual computing nodes in the data center.

The analysis of scientific sources on the topic of the study has shown that:

a) so far, there are no effective algorithmic solutions for planning virtual machines and containers with virtual network functions, cloud services, application-oriented accounting topology of the computer system, and communication tasks schemes;

b) the existing solutions for managing virtual machines and containers on multi-cloud platforms plan computing tasks without subsequent adjustment of network to their communication schemes and use traditional routing methods;

c) the existing methods of data flow routing can be enhanced by taking into account the QoS requirements and distributed nature of a virtual network functions.

This demonstrates the novelty of the solutions offered by the project.

8. Conclusion
We propose an efficient algorithm for launch and deployment of virtual network functions in the infrastructure of a virtual data center. The problem of the optimization of launch and deployment of virtual network functions based on the template VM and containers with disabilities infrastructure virtual data center is reduced to the problem of packing in containers. We also generalize the well known/renowned heuristic and deterministic algorithms of Karmakar-Karp. We have developed an efficient algorithm to place VM by neural network optimization. Comparing the developed algorithm with the exact algorithm, we find that its approximate solutions do not differ much from the exact solutions. Thus, assessing the overall result of the algorithm performance gain can be obtained from 12 to 15% compared with conventional tools that are extremely effective at high intensities requests.

Acknowledgments
The research work was funded by Russian Foundation for Basic Research, according to the research projects No. 16-37-60086 mol_a_dk and 16-07-01004, and the President of the Russian Federation within the grant for state support of young Russian scientists (MK-1624.2017.9).
References
[1] Bolodurina I and Parfenov D 2017 Development and research of models of organization storages based on the software-defined infrastructure Proc. 40th International Conference on Telecommunications and Signal Processing (Barcelona: IEEE Press) p 1–5
[2] Bolodurina I and Parfenov D 2017 Methods and algorithms optimization of adaptive traffic control in the virtual data center Proc. International Siberian Conference on Control and Communications, SIBCON 2017 (Astana: IEEE Press) p 1–5
[3] Bolodurina I and Parfenov D 2017 Development and Research of Models of Organization Distributed Cloud Computing Based on the Software-defined Infrastructure Procedia Computer Science 103 p 569–576
[4] Guisheng Fan, Huiqun Yu, Liqiong Chen, Dongmei Liu 2016 Modeling and analyzing cost and utilization based task scheduling for cloud application Proc. 40th Annual Computer Software and Applications Conference (COMPSAC) (Atlanta: IEEE Press) p 245–250
[5] Ruozhou Yu, Guoliang Xue, Vishnu Teja Kilari and Xiang Zhang 2015 Network function virtualization in the multi-tenant cloud IEEE Network (IEEE Press) p 42–47
[6] Aaron Gember-Jacobson, Raajay Viswanathan, Chaithan Prakash, Robert Grandl, Junaid Khalid, Sourav Das and Aditya Akella 2014 OpenNF: Enabling Innovation in Network Function Control Proc. of the 2014 ACM conference on SIGCOMM (Chicago: ACM) p 163–174
[7] Ying-Dar Lin, Po-Ching Lin, Chih-Hung Yeh, Yao-Chun Wang and Yuan-Cheng LaiAn 2015 An extended SDN architecture for network function virtualization with a case study on intrusion prevention IEEE Network (IEEE Press) p 48–53
[8] Hassan Jameel Asghar, Luca Melis, Cyril Soldani, Emiliano De Cristofaro, Mohamed Ali Kaafar and Laurent 2016 Mathy SplitBox: Toward Efficient Private Network Function Virtualization Proc. of the 2016 workshop on Hot topics in Middleboxes and Network Function Virtualization (Florianopolis: ACM) p 7–13