***Journal of Advanced Computer Science & Technology**, 3 (2) (2014) 232-248*
©Science Publishing Corporation
www.sciencepubco.com/index.php/JACST
doi: 10.14419/jacst.v3i2.3754
Review Paper

A survey on SDN, the future of networking

Shiva Rowshanrad *; Sahar Namvarasl; Vajihe Abdi; Maryam Hajizadeh; Manijeh Keshtgary

Dept. of Information Technology and Computer Engineering, Shiraz University of Technology (SUTECH), Iran, Shiraz
**Corresponding author E-mail: shiva.rrad@gmail.com*

Abstract

Software Defined Networking (SDN) is an emerging architecture which decouples networks control plane and data plane physically. It makes control plane programmable trough a centralized controller, and builds intelligent and flexible networks. The OpenFlow is one of the most famous SDN protocols, which acts as a southbound interface between control plane and data plane. In this survey, SDN implementation approaches and different southbound interfaces, beside different version of OpenFlow, are introduced. In addition to general architecture of SDN, different wireless architectures are discussed. Here, also potential SDN's applications and research areas including hot topics such as Information Centric Networks, Cloud and datacenters, multimedia, wireless and mobile networks over SDN are reviewed.

***Keywords**: Control Plane, Data Plane, Programmable Network, OpenFlow, Software Defined Network.*

1. Introduction

Use of mobile devices, emerging technologies such as cloud computing and virtualization, caused changing of traffic patterns. The rise of Big Data in datacenters arise the need for high network capacity and network scaling. To support these needs, network devices become more complex. Furthermore it would be hard and time consuming for administrators to configure individual devices due to even little changes in network, such as adding or omitting a device. They should reconfigure many multivendor switches and routers, ACLs, which may cause inconsistency and errors [1-3].
The Idea of programmable networks was introduced to meet these challenges and facilitate network evolution. As a result, Software Defined Networking (SDN) is a new paradigm, which revolutionized traditional network architecture. SDN effectively separates the control plane from data plane and move it to a centralized server named controller. It moves the complexity of network management into a software-based controller and provides an abstraction of underlying infrastructure. This allows simple data plane and network devices, makes control plane to be directly programmable and manageable in a centralized manner [1], [4], [5].
The SDN architecture provides programmability, flexibility and reliability over networks. Network operators can implement their own protocols, rules and policies with common programming languages. They can achieve flexible control over network services such as routing, traffic engineering, QOS and security. Network can adapt itself depends on users' requirements. Network management and configurations can be automated through the centralized controller and standard open API, making the network scale easily. By using SDN, administrators are able to add features to control plane without changing data plane or enhance devices in data plane without changing control plane. Decupling control plane from infrastructure is also important because it reduces costs and inconvenience of testing new ideas and strategies in network or deploying new architectures [1], [3], [5], [6]. In this paper, we present SDN architecture, its capabilities, deployment, applications and challenges to give a broader view for those who are interested in this area.
The rest of the paper is organized as follows. Section 2 represents the history of programmable networks. In Section 3 wired and wireless SDN architectures are discussed and different protocols used in these architectures are presented. SDN networks implementation and tests are discussed in section 4. In section 5, the SDN applications and its open research areas are reviewed. Section 6 describes existing challenges of SDN. Finally section 7 concludes the paper.

2. History

Many years ago, the idea of programmable networks, resulting SDN paradigm, has been created [6]. In 1988 SOFTNET [7], in 1990 Active Networking [8], in 1995 OPENSIG [9] and GSMP [10], in 1998, IEEE P1520 Standards Initiative [11], in 2004 4D Project [12] and SoftRouter [13] architecture and finally in 2006 NETCONF [14], Ethane [15] and SANE [16] were proposed to answer the idea of programming network. SANE developed as a prototype, considered a logical server that performed all access control decision. Ethane separated controller and Ethane switch toward providing policy and security by an identity-based accessing. But in 2010 according to RFC 5810 by IETF, ForCES (Forwarding and Control Element Separation) [17] "A parallel approach to software-defined networking" was announced which standardized the communication between controller and network elements. The difference between ForCES and OpenFlow (OF) is their network architectures. ForCES assumed no changing in network architecture so that controller and network elements are situated in one single device as regards to this point that they are apart from each other, but OF separates the controller and network elements physically. Several efforts in the field of SDN have been taken until now. Internet Research Task Force (IRTF), defined The Software Defined Networking Research Group (SDNRG) [18], intended to figure out future SDN approaches and challenges. Also there is a Home provided for SDN researchers in [19].

3. Architecture

SDN architecture consists of 3 layers: infrastructure layer, control layer and application layer. Infrastructure layer or data plane is responsible for forwarding the packets by means of simple forwarders and switches. Control layer contains the controller to manage the Infrastructure layer. In application layer the business applications can interact with network services and capabilities. There is also a need for southbound and northbound interfaces to enable the controller to communicate with the two other layers [1]. More details of SDN components are described in this section.

3.1. Control layer

SDN architecture decouples two parts of a network device. In controller plane, a software program is placed on top and is separated from switches as shown in Fig. 1. Controller plane manipulates the forwarding table for each switch request based on the header of packet-in message (a packet which is created by switch upon table-miss) and sends respond via packet-out or flow-mod message and tracks all applications requests. Each controller plane is built from two components: applications and network operating system. The application part is in many of software programs from metering to monitoring which network virtualization is one of them. When lots of applications regardless of their works confront with WAN, resources become scarce and SDN architecture should be completed with a network packet broker (NPB) and access monitoring [20].

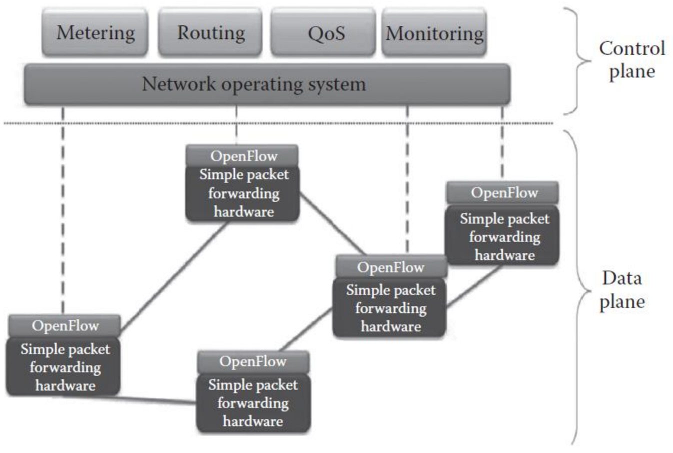

Fig. 1: SDN Architecture [5]

NPB could be one of the controller part that checks the required resources whether it is available or not. So NPB needs to have network topology, past and current traffic loads [20]. Some of controller platforms are compared in Table 1.
Controllers can be deployed in a centralized or distributed manner. One of the most obvious advantages of using a centralized controller is that management and retrieving information would be applied from one logical point (controller) resulting uniform network. Centralized controller like each centralized model has the disadvantages of single point of failure, terminating no availability and scalability. Network could have more than one controller, so each controller is responsible to control a group of network switches, which may interfere with each other, thus one

controller is chosen to be the main controller and the others would be backups [6]. ONIX [21] and HyperFlow [22], suggest a logically centralized but physically distributed control plane. Each switch is controlled with its controller and controllers are distributed in the network, using memory resident (in-memory database) and applications that are written on top of the controllers consider all of the controllers as one central controller. Although in this option lookup overhead will be reduced and scalability and availability are supported, establish consistency and compatibility between network OS and any kind of supported OF and even between controllers are some of the most important issues. Hierarchical view of the controllers like Kandoo [23], manage their related switches and there exists one controller that manages local controllers in order to implement applications on it. Using this case like ONIX and HyperFlow reduces lookup overhead, present availability and scalability with the central view of the whole network without worrying about its consistency and compatibility. Also a proxy controller, FlowVisor [24] is used to set a logical decentralization and give a virtualization view to the controllers in one network.

Packet granularity requires controller plane decision whenever a packet is arrived in data plane. When a table-miss is happened in a switch, switch will encapsulate the entire received packet as a packet-in message and sends it to the controller, decision will be made and a packet-out message will be sent to the switch. In this case, huge amount of data would be passing through the controller and switch that could cause delay. Flow granularity is another selection. When a flow is passed through a switch, for the first time that switch receives that flow, it requires controller plane decision so sends packet-in message which contains only the header of original packet and at the controller side, decision will be made and flow-mod message containing original packet header will be sent to switch. After that, switch will decide based on last decision. In a switch, controller plane decisions would be applied to any kind of flow, subordinates to source, destination, data or any kind of information relates to that flow. If there isn't any decision for huge number of flows received at switch, a lot of connection would be required for any of them; which causes delay. In active policy, switch notifies the controller each time it needs a decision. In proactive policy, controller passes control information to the switch. Both of these policies would be merged to any kind of granularity [6].

Table 1: Different Controllers' Platforms In SDN

Controller	Language	Created by	Open source	OpenFlow version	Description
NOX	Python, C++	Nicira	Yes	1.0, 1.3	Asynchronous, event-based programming model, component based framework. NOX-MT is multithreaded with improving throughput and response time [25].
POX	Python (2.7)	Nicira	Yes	1.0	Component based framework, targets Linux, Mac OS, and Window [26].
Beacon	Java	Stanford university	Yes	1.0.1	Event based and threaded Cross-platform, Dynamic, and Rapid Development [27].
Maestro	Java	Rice university	Yes	1.0	Modular network control applications, multi-thread [28].
Floodlight	Java	Big Switch Networks	Yes	1.0	Based on Beacon, core architecture is modular, open source agent (Indigo) [29].
Floodlight-plus	Java	Big Switch Networks	Yes	1.3	New version of floodlight for supporting OF 1.3 [30].
Ryu	Python	NTT Labs	Yes	1.0, 1.2, 1.3, 1.4	Component based, supporting components development in other languages, event management and reusable NETCONF library, sFlow/Netflow library [31].
OpenDaylight	Java	Linux Foundation	Yes	1.0 , 1.3	Modular, pluggable, and flexible controller platform, supporting multiple southbound protocols [32].

3.2. Infrastructure layer

Infrastructure layer is the SDN switch. Switches in software defined networks are basic forwarding elements which communicate with controllers via an open interface such as OpenFlow. An OF switch consists of at least three main parts: flow table, secure channel and OpenFlow protocol. Flow table(s) is used to lookup packet and also do forwarding. A secure channel is usually a TLS or SSL channel between switch and controller. OpenFlow protocol is for communicating with the switches and managing them. Controller can update, delete, and add the flow entries of flow tables using OF messages. Each flow table consists of many entries, each consists of several components [33-37]. A flow is a sequence of packets that matches a specific entry in a flow table. On receipt of a packet, the switch matches the packet header fields with match fields' component of entries. After table lookup and matching process, the instructions of entry with higher priority will be done on the packet. The fields from packets used to match with flow entries are shown in Table 2.

Table 2: The Fields from Packets Used to Match with Flow Entries

Ingress port	Ether source	Ether destination	Ether type	VLAN ID	VLAN priority	IP Src	IP Dst	IP protocol	TOS	TCP Src	TCP Dst

There are two types of OpenFlow switches: OpenFlow-only, and OpenFlow-hybrid. OpenFlow-only switches support only OF operation, in those switches all packets are processed by the OpenFlow pipeline and the forwarding decisions are made by controllers. OpenFlow-hybrid switches support OF operation in addition to traditional operation such as L2 Ethernet switching, L3 routing etc. A classification mechanism is needed to classify the traffics of OF pipeline and normal pipeline. This mechanism may use VLAN tags, Input ports of the packets, etc. [33-37]. In another aspect, the switches can be classified into Hardware based switches and software based (virtual) switches. Table 3 lists some of the currently SDN available switches.

Table 3: Available SDN Switches

Switch	Type	Series/ Versions	OpenFlow version
Arista [38]	HW	7050 ,7150, 7500	1.0
Brocade [39]	HW	CES 2000, CER 2000, MLX	1.0, 1.3
HP [40]	HW	3500, 3500yl, 5400zl, 6200yl, 6600	1.0, 1.3
IBM [41, 42]	HW	IBM 8264, RackSwitch G8264, G8264T	1.0
LINC[43]	SW	-	1.2,1.3,1.4
NEC [44]	HW	PF5240, PF5248	1.0 , 1.3.1
Open vSwitch (OVS) [45]	SW	Latest version OVS 2.1.2	OpenFlow 1.0 for OVS 1.9 and earlier, OpenFlow 1.2 and 1.3 for OVS 1.10 and later with some missing features, OpenFlow 1.1 for OVS 2.0 and later with some missing features, experimental support of OpenFlow 1.4 for OVS 2.2
Pica8[46]	HW	P-3290, P-3295, P3930 , P-3297 , P-3922	1.0, 1,1, 1.2, 1.3, 1.4

3.3. Northbound interface

SDN Applications are programs that may consume an abstract view of the network for their decision making goals. Moreover, these applications programmatically and directly convey their network requirements and desired network behavior to the SDN Controller. Interfaces between SDN Applications and SDN Controllers are known as Controller-Application Interaction or SDN Northbound Interfaces (NBIs). Northbound Interface of component can be introduced as an element that conceptualizes the lower level details of functions used by component. This interface is usually positioned at the top of the corresponding component, which is the basis of the "northbound interface" title [47].
Despite southbound interface that is well defined in protocols such as ForCES and OF, no comprehensive standard is defined for northbound interface and they are more likely to be developed for particular SDN applications [6]. One justification is that the southbound must enable hardware implementation, while northbound interface's definition completely in software.
For various reasons controllers may need to communicate with each other, on the other side, network applications may require extraction of information about the underlying network policy aspect, then there should be a clearly defined interface. There are some proposals such as Procera [48], Frenetic [49], FML [50] and Nettle [51] which build a policy layer by using a network configuration language. In addition, the northbound API must authorize employment of different policies to the same flow. The modularization approach proposed by [52] tries to avoid rules installed for one task override other rules. This goal is achieved by implementation of an abstraction layer.
IETF with the goal of network traffic engineering deployed Application-Layer Traffic Optimization (ALTO) [53] which currently has been considered for content delivery networks (CDNs) implementation. According to information provided by ALTO, server network was optimized that improve performance and resource consumption. This information gathers by mapping network topology and create abstract and logical model of network by server. Exchanging request and response is accomplished thereby JSON [54]. The set of ALTO features convert this API to appropriate option for SDN implementation[4].
Interface to the Routing System (I2RS) working group is another API for northbound was developed by IETF that its objective was to provide standard for programmable network. Policy-based routing can be mentioned as important trait of I2RS. This interface in compared with similar protocol such as SNMP and NETCONF. It is more rapid and application-friendly[4].

3.4. Southbound interface

In SDN, the most popular southbound interface is the OF protocol. OF enables communication between controller and the nodes in the network, so the controller discover network topology, get reports from the nodes, instruct and manage them as needed and implement requests relayed to flows via northbound APIs. There are also other southbound interfaces, which are described in this section [5], [55].

3.4.1. Openflow protocol

OF protocol is the most well-known interface between forwarders and controllers in SDN. This protocol is defined for Ethernet-based networks. OF protocol have different versions. The first version was OF 0.2.0 released in March 2008 as a draft. The OF 1.0 specification is the first version which has official vendor support. The latest specification is OF 1.4 which released in October 2013. As shown in Table 3 OF 1.0 and 1.3 are the most deployed versions. Fig. 2 shows the OpenFlow changes through its different versions. More details are described in this section.

OF 1.0 [33] supports a single flow table with flow entries consist of three components: Header Fields, Counters and Actions. The Match field supports 12 header fields which are shown in Table 2. Each flow entry contains a specific value or ANY keyword which matches any value. The entries must associate with zero or more actions that should be done on the matching packets in order. The packets with zero actions or action lists which cannot be processed by the switch are dropped. If no matches found for a packet the first 200 bytes of the packet are sent to the controller through the secure channel.

In OF 1.1 [34] several flow tables are pipelined. When a switch receives a packet, it starts to look for an entry to be matched in table0. If the entry found, the packets get processed according to the entry Instructions. The instructions may contain Goto Instruction which points to another flow table. The flow entries can also point to a Group table which consists of group entries. The Actions in Group table are common for multiple flows. In this version the Match fields have 15 tuples. Metadata filed is added to pass information between tables. Two other fields are added to support MPLS tagging. It also uses Instructions instead of Actions component which are more complex and include modifying the packets. Other added supports are: Support for VLAN and QinQ, adding, modifying and removing VLAN headers, Support for virtual ports and tunnels, Multipath routing and ECMP.

OF 1.2 [35] adds support for major features like IPv6 and extensible matches using TLV structure. Using TLV structure, cause more flexibility, since in previous protocols the order and length of match fields was fixed. With OF 1.2 the switches can connect to multiple controllers concurrently. This allows for a better failure recovery and also load-balancing between controllers.

In OF 1.3 [36] every flow table must have a table-miss entry which indicates how to process a packet that has not any match in the table. It may instruct to drop the packet, send it to the controller or direct the packet to another table. Another improvement over previous versions is meter tables which make OF to implement QOS operations like rate limiting. Also a specific duration filed is added to most statistics which helps to compute rates. A cookie field is added to messages containing packets sent to the controller. This helps controller, process the messages faster than if it had to search its entire database.

In Previous version of OF, a switch could connect to multiple controllers for fault tolerance and load balancing. OF 1.3 introduces per connection event filtering which improves the multi-controller support by enabling each controller to filter events from the switch it does not want. It also enables a switch to create auxiliary connections to supplement the main connection between the switch and the controller. This allows using the parallelism ability implemented in most switches.

OF 1.4 [37] is the latest specification. In this version TLV structures is used in more parts, improving extensibility. Support for optical ports is added by means of new set of port properties. They include fields for configuring and monitoring transmit/receive frequency and power of a laser in either Ethernet optical port or optical ports on circuit switches.

In previous versions, when a flow table is full, the switch sends an error message to controller to operate on the flow table. This is time consuming and may cause problem. In OF 1.4 an Eviction option is added which makes the switch able to eliminate lower important entries automatically and make space available for new entries. Switches can also inform controller of tables getting full (based on a capacity threshold chosen by controller) via vacancy events and avoid tables to get full. Bundle mechanism is added for quasi-atomic execution of a group of instructions. The Error message is sub-type of symmetric messages in this version and can be initiated by either switch or the controller.

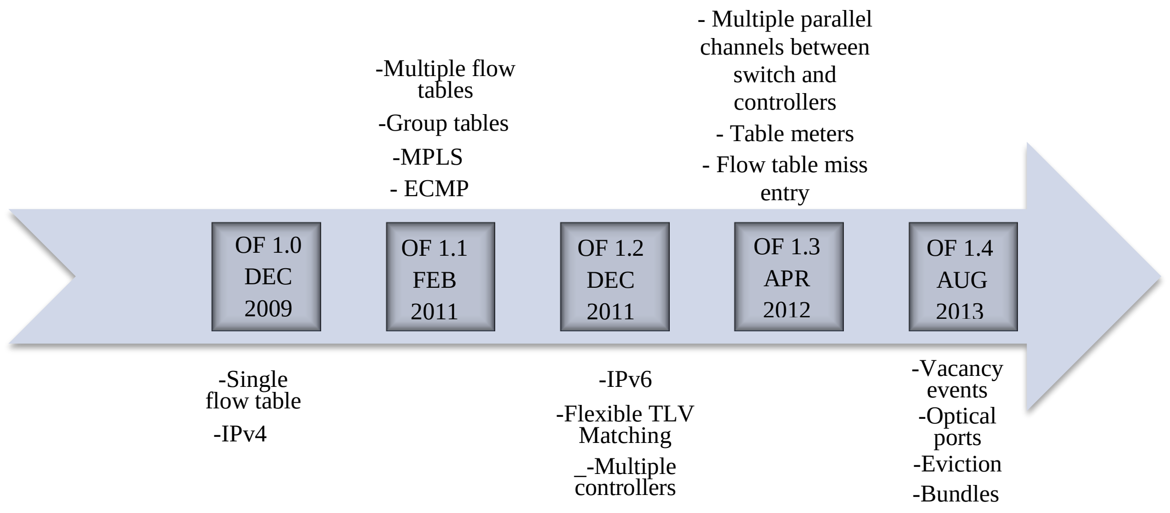

Fig. 2: Open flow Revolution

3.4.2. XMPP

The Extensible Messaging and Presence Protocol (XMPP) [56] is a XML-based protocol introduced by Internet Engineering Task Force (IETF). The basic idea was proposed by Jabber community in 1999 and they introduced an open-source protocol. Afterward IETF represented the improvement version for instance message in 2002. Finally in 2004 the standard definition was published on RFC3920 and RFC3921. XMPP enable exchange “XML stanzas”, small piece of XML, between different entities based on client-server architecture. All connection of this protocol is performed by TCP. Decentralized system of XMPP enables everyone to implement their own XMPP servers and launch their domain. At Juniper Networks Contrail [3], an open source solution introduced by Juniper Network Company was intended XMPP as southbound protocol and was implemented in the controller. As a result, it is possible to build high scalable virtual network and control cloud services automatically.

3.4.3. OnePK

One Platform Kit (OnePK) [4] is API & software development kit (SDK) toolkit to develop controller application for programming Cisco network devices managing and planning. OnePK consists of three main elements: presentation Layer which provides required library for programmers and supports multi language programming such as C, Java and Python, API Infrastructure with task of coordination among various platform and Communication Channel that put out security and flexibility between controller and network elements. Provided service sets enable network administrator to manage and control great number of elements.

3.5. Wireless architectures

We described the general architecture of SDN in previous sections which is mainly used for wired networks. But SDN also can be deployed for wireless networks. Currently, researchers’ main focus in SDN wireless architecture, is centralized control of wireless networks: A central controller's main duties such as management wireless access points, user verification, etc. [57]. In comparison with distributed wireless networks, the centralized system has better consistency because of global view of the network controller. One other feature of wireless SDN networks is that they present peculiarities because its control requires the knowledge and policy about the radio interference and the node mobility [5].

3.5.1. Openroad

For controlling wireless SDNs one of the first approaches was OpenRoad [58]. This open source platform was developed to specify the wireless OF over NOX. OpenRoad architecture [5], is composed of three layers as illustrated in Fig. 3, permits expanded modules to monitor the wireless medium by employing simple network management protocol (SNMP). Flow layer's main responsibility is management of the flow tables in the access points. Network virtualization is the main duty of Slicing layer. FlowVisor in this layer responsible for sharing the OpenFlow tables in the switches. Purposed architecture also required SNMP demultiplexer module in this layer to performing the network slicing for the SNMP traffic, because FlowVisor did not slice the access through SNMP. The last layer is Control layer which represented the OF control plane. Finally, OpenRoad has a centralized controller that provides resource virtualization and different policies can work to gather.

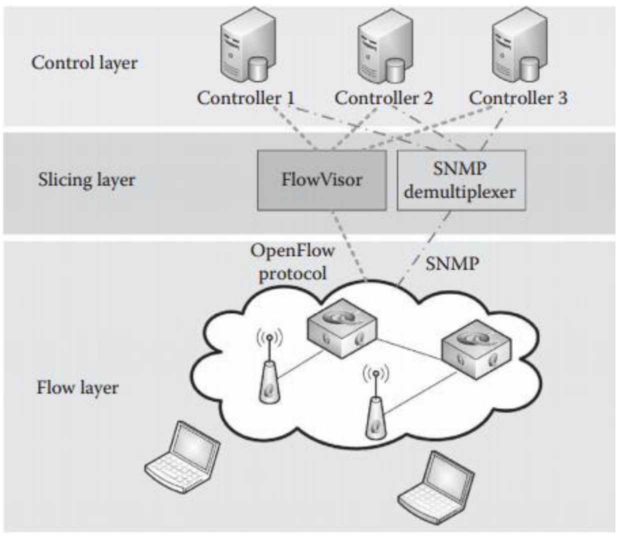

Fig. 3: Open roads Architecture, Which Consists of Three Layers [5].

3.5.2. SoftRAN

Nowadays, distributed algorithms are utilized by radio access network in order to handovers management. While the decision making in environment with few base stations can be easy, it will be relatively harder to swiftly select the best candidate in dense deployments of base stations. For the purpose of handling increasing mobile traffic, some researchers offered SoftRAN [59], a centralized software defined radio access network designed for performing handovers efficiently. In SoftRAN all the base stations are abstracted and controlled in a centralized way. In order to manage every base station, control plane used defined APIs to communicate with radio elements.
General architecture of SoftRAN is illustrated in Fig 4. It periodically gathers the states of base stations and updates the global view of the network. The controller modules require that information to manage radio resource, so collected in the form of a database.

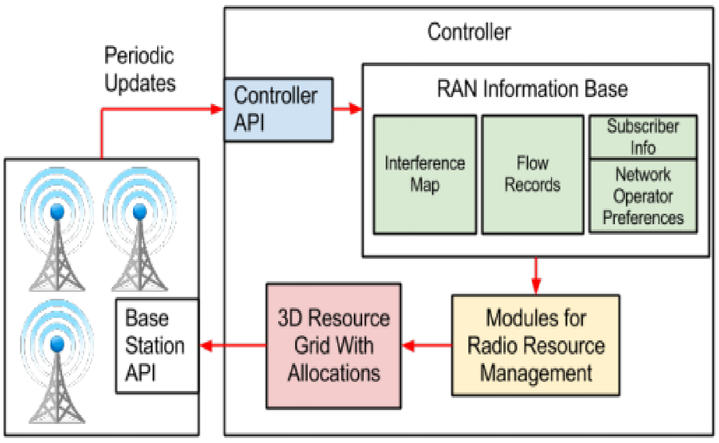

Fig. 4: Architecture of SoftRAN [59]

3.5.3. SoftCell

For solving significant scalability problems exist in cellular networks, Li Erran Li et al. [60] proposed a cellular SDN architecture with local control agents with ability to make simple decisions, as illustrated in Fig. 5. The centralized controller responsible for interpreting flows with high level abstractions. On the Other side, some actions and policies are run on a cell agent for improving the performance and efficiency of the centralized controller.

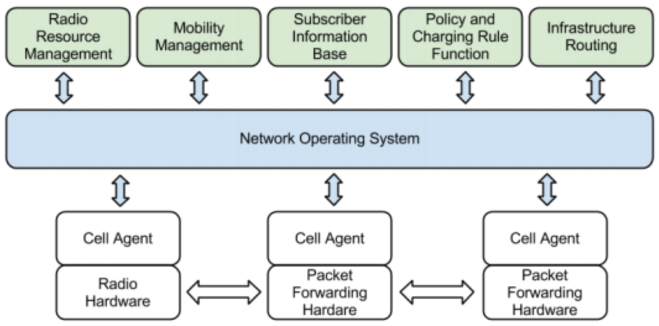
Radio Resource Management
Mobility Management
Subscriber Information Base
Policy and Charging Rule Function
Infrastructure Routing
Network Operating System
Cell Agent
Radio Hardware
Cell Agent
Packet Forwarding Hardare
Cell Agent
Packet Forwarding Hardware

Fig. 5: Cellular SDN with Local Agent [60]

4. Implementation and testing

As mentioned in previous section, many switches and controllers are available for implementing a software defined network. But there is also ways for testing ideas in SDN such as using simulators, emulators and test-beds. NetFPGA's [61] can be used for building a SDN node. Debuggers are also available for testing controllers.

4.1. Simulators and emulators

In this section available SDN simulators and emulators such as Mininet, NS-3 and Estinet are described and compared.

4.1.1. Mininet

Mininet [62], an emulator platform using OF protocol, runs a collection of end-hosts, switches, routers and links on a single Linux kernel by using lightweight virtualization. Components of Mininet act as real network components. This emulator has lots of tools to check the possible bandwidth, the connectivity among nodes and deepest nodes, and the speed of flows. These tools are named Ipref, Ping, PingAll, PingPair, CBench and also Wireshark in order to view network traffic. Mininet is used by developers, teachers and researchers and this is because of easily interaction with network using CLI and API, customizing and sharing features and also development feature on real hardware. It should be mentioned that Mininet is actively developed and supported. Mininet is used widely just because of: fast to start a simple network, supporting custom topologies and packet forwarding, running real programs available on Linux, running on laptops, servers, virtual machines, having sharing and replicating ability, easy to use, being in open source and active development state. In contrast with these advantages, Mininet has some disadvantages too: having disability of transferring huge amount of data in one single system, non-available supporting arbitrary OF controllers, supporting just one platform (Linux kernel), doing NAT out of box, having sharing host file system and PID space and virtual time notion absence. Measuring performance is presented in [63].

4.1.2. NS-3

NS-3 [64] is a discrete event network simulator which is suited for researchers and educators. The NS-3 library is split across many modules organized under the modules tab. One of these modules is OF conformable to SDN. NS-3 has OpenFlowSwitchNetDevice object behaves as a switch and is OF compatible. This object implements a flow table for all received packets and also a connection to controller just like SDN architecture. Two controllers are available in original package, DropController and LearningController (based on learning switch algorithm) [65]. This simulator has these advantages: adding new protocols, shortage of distance among real network and simulated network, having integration and customizable without remaking the core of simulator. NS-3 disadvantages are: loss of available models, absence of visual interface for creating topology and visible capability in experimental level.

4.1. 3 EstiNet

EstiNet 8.0 [66] simulator and emulator supports many OF 1.3.2 and 1.0.0 switches. Besides this advantage, in the simulation mode of EstiNet; POX, NOX, Floodlight and Ryu controllers will have the role of SDN controller plane. In the emulation mode of EstiNet, these controllers can run up on an external machine that is different from the machine which simulates switches. Also in this mode implementation of the controller as a dedicated hardware device using an Ethernet cable is possible, resulting remote controlling. The other benefits of using this simulator/emulator are: accuracy, quickness, repetition and scalability. EstiNet has a unique capability which is called "kernel-reentering simulation methodology". By using this, testing of a novel controller applications program by an OF researcher is easy and effective. A comparison between Mininet, NS-3 and EstiNet is presented in Table 4.

Table 4: Comparision of SDN Simulators and Emulators

Simulator /emulator	Open source	Language	Platform	License	OpenFlow version	Docs
Mininet (Emulator)	Yes	Python	Ubuntu or experimentally Fedora	BSD open source	OF 1.3 of the reference user switch and NOX from CPqD and Ericsson	Good
NS-3 (simulator)	Yes	C++, Python	Linux, Mac, Free BSD	GNU GPLv2	Pre OF 1.0 and version of OF-SID that support MPLS	Good
EstiNet (emulator/simulator)	NO	-	Linux	-	OF 1.3.2 and 1.0.0	Medium

4.2. NeTFPGA

NetFPGA is the software and hardware platform aim for researchers and is used for teaching. Now it is available in 2 platforms: NetFPGA-1G (1G) and the NetFPGA-10G (10G). Due to its open source software and low cost hardware, it is taken into consideration of many students. NetFPGA consists of PCI with four Ethernet port, Xilinx Virtex II pro, static RAM and Double Date Rate (DDR2) SDRAM [4].
Based on advantages that mentioned and flexibility of NetFPGA platform, it is intended as one solution to implement SDN. Field Programmable Gate Array (FPGA) allows to program core processing with user defined logic and embedded core allows to program control functions[61].
In [67] Pang-Wei Tsai et al. described a solution to integrate NetFPGA and OF into network emulation test-bed. Therefore, the authors succeed to implement traffic-controllable network test-bed with high speed packet processing.
J. Naous et al. presented a paper and explained the result of implementing OF on NetFPGA in [68].They stated that this implementation is full line-rate and easy to monitor network flow.

4.3. Test-beds

OF test-beds are experimental environments that mainly developed for testing novel applications using OF. Currently, there are some open test-beds that can be employed for academic research. In continue, some of the most important OF test-beds are introduced and described.
GENI: GENI [69] stands for Global Environment for Network Innovations that is supported by the National Science Foundation. It is a wide suite of infrastructure, developed to support experimental research in networking by creating a huge test-bed [5] In order to reach its goal, GENI federated some platform such as PlanetLab [70], Internet 2 [71], Emulab [72], etc.
At a high level, it has the prospect of experimental test-bed in which all components will be programmable, federated and virtualizable, thus it is the best candidate when the deployment scales [73]. Nowadays, GENI project in conjunction with Internet2 are enabling a SDN based network ready for the deployment novel architecture for new network services and future internet [74].
OFELIA: Funded by Seventh Frame-work Program (FP7) of European Union. The idea behind OFELIA [75] is a test-bed in which researchers can dynamically control and extend the network via OF. The OFELIA infrastructure formed by a set of interconnected islands, each of them managed independently. Employment of virtualization, permits each experimenter receives a network slice. Each slice is formed by virtual machines to run the clients and server, virtual machine which corresponds to a typical OF controller and virtual OF network. OFELIA Control Framework (OCF) is a Common control framework to performed experiment. OCF provides tools for user verification and access, allocation of the slice, configuration of the network [5].
FIBRE: The main goal of FIBRE is to create common space between Brazil and Europe for future Internet architectures [76]. The key idea behind this project [77], is to build a federated test-bed by combining wireless networks and wide area network. The wireless network and wide area network are controlled using OF[5].

4.4. Debuggers

As mentioned before SDN controller is programmable and this feature increases the probability of inadvertently errors. Generally, finding bugs is hard and time-consuming therefore debuggers have become one of the important components of OF/SDN. Debuggers are tools that are used to test and diagnose program and enable programmers to interact with program while it is executing on computer. OF debuggers allow us to trace packet flow behavior to check whether the network is operating as expected[5]. Nikhil Handigol et al. in[78] introduced ndb, a SDN debugger. Their idea was similar to gdb, a debugger for GNU operating system, which changes original control flow without any changes to its semantic. The main goal of the authors was using familiar action specially breakpoint. Breakpoint operation calls all functions that lead program to specified point.

A.Khurshid et al. presented VeriFlow in [79], a layer between controller and hosts that analyzes and checks network configuration to find bugs without having negative impact on network performance. They claimed that implementation of VeriFlow is compatible with NOX controller and demonstrated that it is able to verify the control plan rules in real time and prevent error to impact the proper functioning of the network.
FlowChecker is a configuration analysis tool which introduced by in [80]. Using this system provides the possibility of checking correctness of data plan configuration and validating consistency of different devices on OF network by interpretation intra/inter-switches flow table using Binary Decision Diagrams. The authors showed that configuration analysis can be done while network is running. This feature is useful to determinate QoS.
M.Kobayashi et al. at [81] presented approach for collecting network parameters and used that data to monitor, analyze and debug SDN. The main information for debugging is gathered from flow table, different traffic statistics and controller messages. Some different tools were used for debugging purpose and checking traffic between controller and switches using Wireshark integration, which was stated as the most effective approach.

5. Applications and research trends

In this section we introduce SDN applications and recent researches including Software defined ICN, Multimedia, network management, network virtualization, cloud and datacenter, security, wireless and mobile networks.

5.1. Software defined ICN

In recent years many researchers claimed that current internet architecture is not able to response the emerging and future need of users. Based on this claim, new architectures were introduced. Information centric network is one of these architectures. In ICN, the information name is unique and independent of locations, applications, storages and distribution and network primitives are done based on the names. To retrieve named information, various transmission techniques are introduced, including name-based routing, name-based resolution, and etc. To support these techniques and exploit the advantages of ICN, dramatic changes to the network devices deployed in current Internet are needed, which leads to challenge of ICN implementation[82], [83]. A number of projects [82-86] proposed implementing ICN over SDN. This leads to decreasing in implementation costs. It also enables innovation and optimization of network resources and functionalities. Proposed methods are discussed in Table 5.

Table 5: Software Defined ICN Implementation Methods

Method Description	Publication	`Description	Implementation
ContentFlow	Carvalho et al. [87]	Changing OF protocol	No implementation
Wrapper/ Proxy	Nguyen et al. [82, 88]	Using wrapper to communicate between OF switch and CCNx (an ICN implementation reference) so the wrapper, OF switch and CCNx act as an ICN router together	OF 1.0
Hash functions	Ooka et al. [86]	Hashing the names into fields (e.g. IP address) that can be processed by OpenFlow	OF 1.0
CoNet	Blefari-Mellazzi et al. [83-85]	Using IP option header as name filed	OF 1.0 and Floodlight controller, Implemented in OFELIA [75] SDN test-bed

Integrating ICN and SDN can help emerging technologies grow efficiently and easily face the challenges in today's networks. Supporting mobility and cloud computing are two controversial subjects in today's networks due to inconsistency with IP protocol. SDN can be used to make the best and optimized decisions for VM migration based on the controllers' view of network and automatically update the network devices configurations after migration [5].
One of the advantages of implementing ICN over SDN is to improve ICN's functionalities. In [89] a routing protocol is proposed which supports mobility by means of controller. This can be easily implemented using software defined ICN. In [90] the authors, discussed the role of virtualization in NDN (an architecture based on ICN) and outlined traffic optimization, traffic engineering and in-network catching management as the advantages of implementing NDN over SDN, specially for multimedia traffics.

5.2. Multimedia and QOS

Today’s internet architecture is based on end-to-end communication control which enables best effort services. This is valuable for data transmission but not for multimedia traffics. Multimedia applications such as video streaming, video on demand, video conferencing, WebTV and etc. require steady network resources and tolerate special amount of delay, jitter and error-rate. Providing these QOS requirements needs to select optimum path among all paths available in the network[91], [92]. QOS architectures like IntServ [93] and DiffServ [94] are proposed for this purpose, which have difficulties. They have lack of a centralized view of the network and choose the path in a hop-by-hop manner which may not be the optimum path. They also need specialized software and hardware requirements for implementation.
Software defined networks make it possible to select different paths for different traffic flows by use of different routing protocols (performing routing prioritization), based on flow’s requirements and a centralized view of the network. Moreover, OF has some QOS features. In OF 1.0 the flows’ packets can be queued in output ports. This queuing mechanism can be configured through protocols such as SNMP, CLI and NetConf [95]. By use of an auxiliary configuration protocol named OF-Config [96] maximum and minimum transmission rate of queues can be configured. OF 1.1 added support for rewriting ECN (Explicit Congestion Notification) bits. The most important feature is table meters which have been supported since OF 1.3. This feature enables rate limiting. Many projects [91, 92, 97, 98] proposed enhancement or optimization methods for multimedia QOS over SDN. A summary of these projects are presented in Table 6.

Table 6: Multimedia and QOS Projects over SDN

Project	Application	Features used	Implementation
Civanlar et.al [91]	Scalable video streaming	Routing prioritization	OF 1.0 , NOX
Kim et.al [97]	Network QOS control	Rate limiting and queuing	OF 1.3, NOX
Owens and Durresi [98]	Video	QOS-routing, Queuing and rate limiting(traffic shaping) – an IntServ like scheme	OF 1.3.1 (simulation)
Egilmez et.al [92]	Video	Routing prioritization (dynamic QOS routing)	OF 1.0 , Floodlight Controller

5.3. Network management

SDN caused an abstract picture of network for simplified policy enforcement and network management so network management is applied from one logical point. OF architecture, as a part of SDN architecture, made network management more flexible, control in the packet or flow granularity levels. Management in software defined network from a centralized logical point upon flow-table at controllers and using that flow-tables distributed in network (switches) cause a flexible network management [99]. Here are some efforts accomplished in the field of network management.
Network configuration because of A) high-level policies specifications in a distributed low-level configuration (using CLI) mode and B) mutable network state is difficult [99]. Authors also in [99] authors solved three problems SDN network management: frequently changes happening in a network, using a high level language for network configuration, fault and error recognition and troubleshooting. They did it by Procera, an event-driven control framework based on SDN paradigm which is based on functional reactive programming (FRP) and OF 1.0.0. High level policies could be translates into a set of rules and four control domain is in operators’ hand: time, data usage and authentication status and traffic flow. Another control system is integrated network management and control system (I-NMCS) discovery and fault detection [100]. Provisioning and control are combined together and is modular, loose coupling, with low overhead and extensible. I-NMCS is used in a hybrid mode which means that only selected flows will be handled by the controllers and rest of the flows will be exerted by network protocols. This is because of authors’ thought that traditional network management and SDN in future networks would be co-exists with each other. Security management is another aspect of network management. Some applications implemented in controllers brings complex processing more than security. In [101] an algorithm of information security management system combined with fuzzy logic and a prototype of intrusion detection system (IDS) is presents. Using fuzzy logic in this paper decreases number of code lines 20-30%. For event-based network control approach, Lithium [102] system and for minimal visibility of performance and control, a programmable measurement platform approach resulting BISmark [103] are offered. In [104] an online configuration management problem which uses OF and includes agility, accuracy, reliability and scalability is presented. This framework is consists of two parts: disaster event detection and filtering segment and disaster correlation and detector management segment. More efforts are doing in network management fields from performance to security.

5.4. Network virtualization

Network virtualization is one of the important research areas of today's network that enable users to share resource and infrastructure. SDN due to central controller is discussed as a solution to manage virtual networks [5].

In [105] J. Matias et.al presented the EHU OF Enabled Facility (EHU-OEF), a system for managing virtual networks and sharing infrastructure based on layer 2 and OF. The authors were selected L2PNV for implementation their system. The benefits of using layer 2 prefix-based network virtualization (L2PNV) for network virtualization consist of ease of configuration and flexibility of header fields that allows users to create different slice and different virtual network. R. Nejbati et.al in [106] was proposed cross-layer approach for network isolation based on OF to optimize Network virtualization without interfering between different slice so users are enable to modify the behavior of own virtual network. The suggested solution was based on FlowVisor and SCML. Slice Control and Management Layer (SCML) is used for management and monitoring slices infrastructure operations.

I. M. Moraes et al. in [107] suggested FITS, a test-bed for completely isolated virtual network with different QoS. FITS provided the possibility for researchers to choose Xen [108], hypervisor for virtualization with the best performance and high ability to manage resource, or OF to test and perform their experiments. Measurement tools of FITS monitor the state of physical and virtual network that helped administrator to manage and allocate resource optimally.

5.5. Cloud and data center

One area that SDN has been attended a lot is Cloud Services and data center. One of the main characteristics of cloud is that users gain the adequate resources based on requirement in real time[109]. Cloud management is the most important challenge that has always been and many solutions have been proposed for that. SDN is highly regarded as one of the newest solutions, which makes it possible to configure and manage cloud and data center easily. In [110] Anderson et.al implemented cloud data center using OF and they found that SDN based implementation is faster and easier to configure because of SDN centralized controller and abstraction management plan. The other challenge of cloud system is maintain costs, that SDN due to centralized location management solve this problem well [5, 109].

Oracle SDN is an example of system that virtualizes data center using definition connectivity in software. This system connects virtual machine to other virtual machines and network servers with 70 percent simpler infrastructure. Using Oracle Fabric Manager's interface configuration and monitoring virtual network is possible from any location. It is claimed that the cost of Oracle SDN system compared with legacy network is 50 percent less where as you benefit of up to 80 GB/s server-to-server throughput [111].

Raghavendra et.al in [112] introduce NetGraph, a software architecture to manage SDN based cloud systems. This system is shared graph library and provides a module with a set of API for monitoring and diagnostics cloud system.

Hedera which is presented by Al-Fares et al. [113], define Dynamic Flow Scheduling for Data Center Networks using OF. A central scheduler makes it possible to balance load based on active flow state on whole system.

In [114] propose cross stratum resilience architecture for OF enabled data center interconnection. Deployment of virtual machines to server is utilized based on service type. Managing resources, degree of their occupation and backup mechanisms are used for resource maintenance. This system is designed for Flexi-Grid optical networks. Hui Yang et.al argued that system has improved end to end responsiveness and optimized resource usage.

5.6. Security

Due to centralized architecture of SDN, it used to detect security problem. Supervision of SDN on whole network flow and monitoring behavior of users makes SDN possible to detect attack rapidly and prevent more damage. Many researchers introduced mechanisms to detect Dos and DDos attack -because of flooding based behavior of them. The example of such system is introduced by Junho Suh et.al in [115] that is implemented on NetFPGA-OF platform.

Chu et al. in [116] introduced system which uses OF switch on edge to pass only safe traffic which is defined on flow table and LISP (Locator/ID separation protocol) to recognize trusty user

In [117] have been explored solutions for scanning-based attack. In that article J. Jafarian et.al introduced OF Random Host Mutation (OFRHM) system which changes real IP of each host with the random virtual IP. With this mechanism fake IP is easily detectable and system prevents large percentage of attacks such as stealthy scanning, worm propagation and etc. OF responsibility is assigned the virtual IP to host frequently.

Considering policy based security is another security solution that Xiong Liu et.al in [118] have dealt with. They introduced multilevel security system that prevents information of specific level which gathered by same or lower level host. In designing system is used OF is used to monitor packet and check content so filter packet with security problem.

Main idea of Y. Jubaa et.al paper in [119] is to isolate network device dynamically from intra-LAN attack using OF. In suggested architecture IDS component is responsible for detecting attacks and recognizing unsecure devises and inform OF controller. They argue that results have shown this architecture is effective enough on actual LAN.

One problem of SDN approach anomaly detection is that when network traffic is high gathering full data from flow table is not efficient so K. Giotis et.al in [120] suggested solution to optimal real time detection system. They combined sFlow [121] and OF and used sampling instead of full flow information and benefited entropy-based algorithm to detect and mitigate anomaly. The authors stated that in low traffic situation performance of system is comparable with native OF architecture.

5.7. Wireless and mobile

SDN can be applied in wireless sensor network [122]. Generally, using SDN in WSNs provided the SDN benefits such as flexibility, easier management, optimized resource utilization, etc. The network controllers have the power to set policies to support several applications by utilizing sensor based software defined wireless network. Also this approach would permit using the same sensor nodes for several applications.
OF can be used for applying flexible control in Wireless Mesh Networks [123]. This approach benefits both features of Mesh networks and OF which are self-configuration and flexible forwarding, respectively.
To apply concepts of abstraction to wireless ad hoc network of smartphone, software defined networking in ad hoc networks [124] was developed. This Hybrid platform has been implemented on Android operating system. The purposed platform is more modular and easier for modification and extension of its components.
Other use case were mentioned in [57] referred the benefit of based SDN, such as Inter-cell interference management and Mobile traffic management. OF could provide seamless handover, dynamic resource management for wireless backhaul, power consumption optimization in mobile backhaul network, Security and Backhaul Optimization and etc. [125].

6. Challenges

Software defined network and OF have confronted with some challenges. Security needs to be everywhere within SDN: A) architecture and its controller, applications, devices, channels (TLS with plain text) and flow table, B) services (to protect availability), C) connected resources and D) information. Also a robust policy framework in order to check and balance controllers, a recovery, report policy and security deployment is still very much up for grabs. The solutions should be simple to deploy and maintain, cost effective and assuredly secure. A new category called software-defined security (SDSec) which is an example of network functions virtualization (NFV) delivers network security enforcement by separating the security control plane from processing and forwarding planes [126].
Besides security, availability (controllers' existence), flexibility [5], [15], [55], [127], controllers and applications compatibility, link and controller reliability are considerable issues. A centralized controller could recover itself in some processes by using backup flows in a way that is not as faster as it is expected [55].
Capital and operational expenses called CAPEX and OPEX is another challenge debated by OF adapters. Availability and reducing system bottleneck would increase CAPEX, however adapters believe that using software defined network and OF reduce the CAPEX. OPEX in SDN would be decreased by diminishing the number of human based configuration, time and error prone fields [55].
Besides these challenges there are some implementation issues for example having 40+ matching fields in a flow, several tables and their large number of flow entries, instructions and actions, flow level programming and controllers' own way programming that must be considered. Also lack of standard APIs in case of overlapping domain among controllers, necessity of encryption APIs for data plane packets, injection APIs for packet and instantaneous APIs for services like IDS and firewall on a switch, absence of operations in case of absence of controller, existence of other packet format are regarding too [128].

7. Conclusion

In this survey, we provided an overview of software defined networks. We introduced programmable networks such as SOFTNET, SANE, ForCes, etc. which resulted the SDN paradigm. Then we described the SDN architecture in three layers: control layer, infrastructure layer and application layer. We discussed about the control layer and its different control platforms such as Floodlight, RYU, OpenDaylight, etc., the infrastructure layer, available switches and their capabilities. Also the northbound interfaces and southbound interfaces including different versions of OpenFlow protocol, XMPP and OnePK were described. Wireless architectures such as OpenRoad, SoftRan and SoftCell are also explained. We also described different tools for testing SDN. We compared Mininet, EstiNet and NS3 as SDN simulators and emulators. We discussed different hardware and software tools and test-beds for SDN. At last we introduced SDN applications and research trends such as software defined ICN, virtualization, wireless and mobile networks, cloud and datacenters, multimedia over SDN and the works done on security of SDN . We also point out the existing challenges of using SDN.

References

[1] Open Networking Fundation , "Software-defined Networking: The New norm for Networks.", *ONF White Paper*, (2012) , available online: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf, last visit:18.10.2014

[2] B. N. Astuto, M. Mendonça, X. N. Nguyen, K. Obraczka and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", *hal-00825087*,(2013), available online: http://hal.inria.fr/hal-00825087, last visit:18.10.2014

[3] K. Bakshi, "Considerations for Software Defined Networking (SDN): Approaches and Use Cases", *IEEE Aerospace Conference*, (2013), pp:1-9.

[4] T. D. Nadeau and K. Gray, *Software Defined Networks*, CA: O' Reilly, (2013).

[5] F. Hu, "Network Innovation through OpenFlow and SDN: Principles and Design", CRC Press, (2014). http://dx.doi.org/10.1201/b16521.

[6] B. A. A. Nunes, M. Mendonca, X. N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks", *IEEE Communications Surveys and Tutorials*, Vol.16, No.3, (2014), pp. 1617-1634, available online: http://dx.doi.org/10.1109/SURV.2014.012214.00180, last visit: 18.10.2014.

[7] J. Zander and R. Forchheimer, "The SOFTNET Project: a Retrospect", *Electrotechnics, 1988. Conference Proceedings on Area Communication, EUROCON 88., 8th European Conference on*, (1988), pp.343-345, available online: http://dx.doi.org/ 10.1109/EURCON.1988.11172, last visit:18.10.2014.

[8] N. Feamster, J. Rexford, and E. Zegura, "The Road to SDN," *Queue*, Vol.11, No.12, (2013), p.20, available online: http://dx.doi.org/10.1145/2559899.2560327 , last visit:18.10.2014.

[9] A. T. Campbell, I. Katzela, K. Miki, and J. Vicente, "Open Signaling for ATM, Internet and Mobile Networks (OPENSIG'98)," *ACM SIGCOMM Computer Communication Review*, Vol.29, No.1, (1999), pp.97-108, available online: http://dx.doi.org/10.1145/505754.505762, last visit:18.10.2014.

[10] A. Doria, F. Hellstrand, K. Sundell, and T. Worster, "General switch management protocol (GSMP) V3," *RFC 3292*, (2002), available online: https://www.ietf.org/rfc/rfc3292.txt, last visit:18.10.2014

[11] J. Biswas, A. A. Lazar, J.-F. Huard, K. Lim, S. Mahjoub, L.-F. Pau, *et al.*, "The IEEE P1520 standards initiative for programmable network interfaces," *Communications Magazine, IEEE*, Vol.36, No.10, (1998), pp.64-70, available online: http://dx.doi.org/64-70,10.1109/35.722138, last visit:18.10.2014.

[12] The 4D Project,"Clean Slate Architectures for Network Management", available online: http://www.cs.cmu.edu/~./4d/#csamp, last visit:18.10.2014.

[13] T. Lakshman, T. Nandagopal, R. Ramjee, K. Sabnani, and T. Woo, "The softrouter architecture," *Proceedings of ACM SIGCOMM Workshop on Hot Topics in Networking*, (2004), available online: http://conferences.sigcomm.org/hotnets/2004/HotNets-III%20Proceedings/lakshman.pdf, last visit:18.10.2014.

[14] R. Enns, M. Bjorklund, and J. Schoenwaelder, "NETCONF configuration protocol," *RFC 6241*, (2011), available online: https://tools.ietf.org/html/rfc6241, last visit:18.10.2014.

[15] M. Casado, M. J. Freedman, J. Pettit, J. Luo, N. McKeown, and S. Shenker, "Ethane: Taking control of the enterprise," *ACM SIGCOMM Computer Communication Review*, Vol.37, No.4, (2007), pp.1-12, available online: http://dx.doi.org/ 10.1145/1282427.1282382, last visit:18.10.2014.

[16] M. Casado, T. Garfinkel, A. Akella, M. J. Freedman, D. Boneh, N. McKeown, *et al.*, "SANE: A protection architecture for enterprise networks," *Proceedings of the 15th conference on USENIX Security Symposium*, Vol.15, No.10, (2006), pp.137–151, available online: https://www.usenix.org/legacy/event/sec06/tech/full_papers/casado/casado_html/, , last visit:18.10.2014.

[17] A. Doria, R. Gopal, H. Khosravi, L. Dong, J. Salim, and W. Wang, "Forwarding and control element separation (ForCES) protocol specification," *RFC 5810*, (2010), available online: http://tools.ietf.org/html/rfc5810, last visit:18.10.2014.

[18] T. Chown and D. R. Newman, "OpenFlow Experiment in Real-Time Internet Edutainment," *OFERTIE Project Partners*, (2013), pp.1-24, available online: www.ofertie.org/files/2014/02/318665OFERTIE-D7-7-1-Initial-Standardisation-Proposal FINAL.pdf, last visit: 18.10.2014.

[19] Open Networking Foundation, available online: https://www.opennetworking.org/, last visit: 18.10.2014.

[20] C. Alaettinoglu, "Software Defined Networking," (2013), available online: www.packetdesign.com/resources/white-papers/sdn-white-paper.pdf, last visit:18.10.2014.

[21] T. Koponen, M. Casado, N. Gude, J. Stribling, L. Poutievski, M. Zhu, *et al.*, "Onix: A Distributed Control Platform for Large-scale Production Networks," *Proceedings of the 9th USENIX conference on Operating systems design and implementation*, (2010), pp.1-6, available online: static.usenix.org/events/osdi10/tech/full_papers/Koponen.pdf, last visit:18.10.2014.

[22] A. Tootoonchian, S. Gorbunov, Y. Ganjali, M. Casado, and R. Sherwood, "On controller performance in software-defined networks," *USENIX Workshop on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services (Hot-ICE)*, (2012), available online: https://www.usenix.org/system/files/conference/hot-ice12/hotice12-final33_0.pdf, last visit:18.10.2014.

[23] Kandoo, available online: http://www.kandoo.org, last visit:18.10.2014.

[24] Flowvisor, available online: https://openflow.stanford.edu/display/DOCS/Flowvisor, last visit:18.10.2014.

[25] NOX ,available online: http://www.noxrepo.org/nox/about-nox/, last visit:18.10.2014.

[26] POX , available online: http://www.noxrepo.org/pox/about-pox/, last visit:18.10.2014.

[27] Beacon , available online: https://openflow.stanford.edu/display/Beacon/Home, last visit:18.10.2014.

[28] E. Ng, "Maestro: A System for Scalable OpenFlow Control," available online: www.cs.rice.edu/~eugeneng/papers/TR10-11.pdf, last visit:18.10.2014.

[29] Floodlight OpenFlow Controller - Project Floodlight, available online: http://www.projectfloodlight.org/floodlight/, last visit:18.10.2014.

[30] Announcing release of Floodlight with OF 1.3 support, available online: http://sdnhub.org/releases/floodlight-plus-openflow13-support/, last visit:18.10.2014.

[31] Ryu 3.9 documentation, available online: http://ryu.readthedocs.org/en/latest/getting_started.html#what-s-ryu, last visit:18.10.2014.

[32] *Opendaylight*. available online: http://www.opendaylight.org/, last visit:18.10.2014.

[33] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.0. 0.", (2009), available online : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.0.0.pdf, last visit:18.10.2014

[34] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.1. 0.", (2011), available online : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.1.0.pdf, last visit:18.10.2014

[35] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.2. 0.", (2011), available online : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.2.0.pdf, last visit:18.10.2014

[36] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.3. 0.", (2012), available online : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.3.0.pdf, last visit:18.10.2014

[37] OpenFlow Switch Consortium, "OpenFlow Switch Specification Version 1.4. 0.", (2013), available online : https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-spec-v1.0.0.pdf, last visit:18.10.2014

[38] Arista Networks, available online: http://www.aristanetworks.com/en/products/eos/openflow, last visit:18.10.2014

[39] Brocade, available online: http://www.brocade.com/products/all/switches/product-details/netiron-ces-2000-series/index.page , last visit:18.10.2014

[40] HP switches, available online: http://h17007.www1.hp.com/us/en/networking/products/switches/HP_3500_and_3500_yl_Switch_Series/index.aspx, last visit:18.10.2014

[41] IBM, available online : http://www-03.ibm.com/systems/networking/software/sdnveof/ , last visit: 8.8.2014

[42] Juniper, http://www.juniper.net/us/en/products-services/sdn/ , last visit:8.8.2014

[43] Y. Cheng, V. Ganti, and V. Lubsey, “Open Data Center Alliance Usage Model: Software-Defined Networking rev. 2.0”, *Open Data Center Alliance*, (2014), [Online]. Available online: http://www. opendatacenteralliance.org/docs/Software Defined-Networking-Master-Usage-Model- Rev2.0.pdf, last visit:8.8.2014

[44] NEC, available online: http://www.necam.com/sdn/doc.cfm?t=PFlowPF5240Switch, last visit:18.10.2014

[45] OpenVSwitch, available online: http://openvswitch.org , last visit:18.10.2014

[46] Pica8 open networking, http://www.pica8.org/open-switching/1gbe-10gbe-40gbe-open-switches.php, last visit:18.10.2014

[47] R. Kumar, Software Defined Networking - a definitive guide, Smashwords, (2013).

[48] H. K. A. Voellmy, and N. Feamster, "Procera: a language for high-level reactive network control", *Proceedings of the Hot topics in software defined networks*, (2012), pp. 43-48, available online: http://dx.doi.org/10.1145/2342441.2342451, last visit: 18.10.2014.

[49] R. H. N. Foster, M. J. Freedman,C. Monsanto, J. Rexford, A. Story, and D. Walker, "Frenetic: a network programming language", *ACM SIGPLAN Notices*, Vol.46, No.9, (2011), pp. 279-291, available online: http://dx.doi.org/10.1145/2034574.2034812, last visit: 18.10.2014.

[50] N. S. Gude, T. L. Hinrichs, M. Casado, J. C. Mitchell, and S. Shenker, "Practical declarative network management", *Proceedings of the ACM workshop on Research on enterprise networking*, (2009), pp. 1-10, available online: http://dx.doi.org/10.1145/1592681.1592683, last visit: 18.10.2014.

[51] A. V. a. P. Hudak., "Nettle: taking the sting out of programming network routers", *Proceedings of the Practical aspects of declarative languages*, (2011), pp. 235-249, available online: http://dx.doi.org/10.1007/978-3-642-18378-2_19, last visit: 18.10.2014.

[52] J. R. C. Monsanto, N. Foster, J. Rexford, and D. Walker, "Composing software-defined networks", *Proceedings of the USENIX Symposium on Networked Systems Design and Implementation*, (2013), pp. 1-14.

[53] J. Seedorf and E. Burger, "Application-layer Traffic Optimization (ALTO): Problem Statement", *RFC 5693*, (2009), available online: http://tools.ietf.org/html/rfc5693, last visit:16.10.2014.

[54] D. Crockford, “The Application/json Media Type for Javascript Object Notation (json)”, *RFC 4627*, (2006), available online: https://tools.ietf.org/html/rfc4627, last visit:17.10.2014.

[55] A. Lara, A. Kolasani, and B. Ramamurthy, "Network innovation using openflow: A survey", *Communications Surveys & Tutorials, IEEE*, Vol.16, No.1, (2013), pp.493–512, available online: http://dx.doi.org/ 10.1109/SURV.2013.081313.00105, last visit:18.10.2014.

[56] P. Saint-Andre, "Extensible Messaging and Presence Protocol (XMPP): Core", *RFC 6120*, (2011), available online: http://xmpp.org/rfcs/rfc6120.html, last visit:17.10.2014.

[57] Open Networking Foundation, “Openflow-enabled mobile and wireless networks”, *ONF whitepaper*, (2013), available online: https://www.opennetworking.org/images/stories/downloads/sdn-resources/solution-briefs/sb-wireless-mobile.pdf, last visit:18.10.2014.

[58] K. K. Yap, M. kobayashi, R. Sherwood, T. Huang, M. Chan, N. Handigol, and N. McKeown, "OpenRoads: Empowering research in mobile networks", *ACM SIGCOMM Computer Communication Review*, Vol.40, No.1, (2010), pp. 125-126, available online: http://dx.doi.org/10.1145/1672308.1672331, last visit: 18.10.2014.

[59] P. D. Gudipati Aditya, Erran Li Li, and Katti Sachin, "Softran: software defined radio access network", *Proceedings of the ACM SIGCOMM workshop on Hot topics in software defined networking*, (2013), pp. 25–30, available online: http://dx.doi.org/10.1145/2491185.2491207, last visit: 18.10.2014.

[60] M. Z. M. Erran Li Li, and Rexford Jennifer, "Toward software defined cellular networks", *Proceedings of the European Workshop on Software Defined Networking*, (2012), pp. 7-12, available online: http://dx.doi.org/10.1109/EWSDN.2012.28, last visit: 18.10.2014.

[61] J. W. Lockwood, N. McKeown, G. Watson, G. Gibb, P. Hartke, J. Naous, *et al.*, "NetFPGA--An Open Platform for Gigabit-Rate Network Switching and Routing", *Proceedings of the 2007 IEEE International Conference on Microelectronic Systems Education*, (2007), pp:160-161, available online: http://dx.doi.org/10.1109/MSE.2007.69, last visit:17.10.2014.

[62] Mininet: An Instant Virtual Network on your Laptop (or other PC), available online: http://mininet.org/, last visit:18.10.2014.

[63] Introduction to Mininet - mininet/ mininet wiki – GitHub, available online: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet, last visit:18.10.2014.

[64] ns-3, available online: http://www.nsnam.org/, last visit:18.10.2014.

[65] ns-3: ns-3 Documentation, available online: http://www.nsnam.org/docs/release/3.19/doxygen/index.html, last visit:18.10.2014.

[66] EstiNet Technologies, available online: http://www.estinet.com/products.php, last visit:18.10.2014.

[67] P.-W. Tsai, P.-W. Cheng, M.-Y. Luo, T.-L. Liu, and C.-S. Yang, "Planning and Implantation of NetFPGA Platform on Network Emulation Testbed", *Proceedings of the asia-Pacific advanced network*, Vol.32, (2011), pp:1-7, available online: http://dx.doi.org/10.7125/APAN.32.1, last visit:17.10.2014.

[68] J. Naous, D. Erickson, G. A. Covington, G. Appenzeller, and N. McKeown, "Implementing an OpenFlow Switch on the NetFPGA Platform", *Proceedings of the 4th ACM/IEEE symposium on architectures for networking and communications systems*, (2008), pp:1-9, available online: http://dx.doi.org/10.1145/1477942.1477944, last visit:17.10.2014.

[69] GENI, available online: http://www.geni.net/, last visit: 18.10.2014.

[70] PLANETLAB, available online: http://www.planet-lab.org/, last visit: 18.10.2014.

[71] Internet2, available online: http://www.internet2.edu/, last visit: 18.10.2014.

[72] Emulab - Network Emulation Testbed Home, available online: http://www.emulab.net/, last visit: 18.10.2014.

[73] C. Elliott, "GENI: opening up new classes of experiments in global networking", *IEEE Internet Computing*, Vol.14, No.1, (2010), pp. 39–42.

[74] F. de Oliveira Silva, J.H. de Souza Pereira, P.F. Rosa, and S.T. Kofuji, "Enabling future internet architecture research and experimentation by using software defined networking", ", *Proceedings of the European Workshop on Software Defined Networking*, (2012), pp. 73-78, available online: http://dx.doi.org/10.1109/EWSDN.2012.24, last visit: 18.10.2014.

[75] OpenFlow in Europe Linking Infrastructure and Applications, available online: http://www.fp7-ofelia.eu/, last visit: 18.10.2014.

[76] A. A. e. S. Sallent, I. Machado, L. Bergesio, S. Fdida, J. Rezende, S. Azodolmolky, M. Salvador, L. Ciuffo, and L. Tassiulas, "Fibre project: Brazil and europe unite forces and testbeds for the internet of the future", *Testbeds and Research Infrastructure. Development of Networks and Communities*, Vol.44, (2012), pp. 372, available online: http://dx.doi.org/10.1007/978-3-642-35576-9_33, last visit: 18.10.2014.

[77] FIBRE Project, available online: http://www.fibre-ict.eu/, last visit: 18.10.2014.

[78] N. Handigol, B. Heller, V. Jeyakumar, D. Mazières, and N. McKeown, "Where Is The Debugger for My Software-Defined Network?", *Proceedings of the first workshop on Hot topics in software defined networks*, (2012), pp:55-60, available online: http://dx.doi.org/10.1145/2342441.2342453, last visit:17.10.2014.

[79] A. Khurshid, W. Zhou, M. Caesar, and P. Godfrey, "Veriflow: Verifying Network-wide Invariants in Real Time", *ACM SIGCOMM Computer Communication Review*, vol.42, (2012), pp:467-472, available online:http://dx.doi.org/10.1145/2342441.2342452, last visit:17.10.2014.

[80] E. Al-Shaer and S. Al-Haj, "FlowChecker: Configuration Analysis and Verification of Federated OpenFlow Infrastructures", *Proceedings of the 3rd ACM workshop on Assurable and usable security configuration*, (2010), pp:37-44, available online: http://dx.doi.org/10.1145/1866898.1866905, last visit:17.10.2014.

[81] M. Kobayashi, S. Seetharaman, G. Parulkar, G. Appenzeller, J. Little, J. van Reijendam, et al., "Maturing of OpenFlow and Software-Defined Networking Through Deployments", *Computer Networks*, (2013), pp:151–175, available online: http://dx.doi.org/10.1016/j.bjp.2013.10.011, last visit:17.10.2014.

[82] X. N. Nguyen, D. Saucez, and T. Thierry, "Providing CCN functionalities over OpenFlow switches", *hal-00920554*, (2013), available online: https://hal.inria.fr/hal-00920554/, last visit: 18.10.2014.

[83] S. Salsano, N. Blefari-Melazzi, A. Detti, G. Morabito, and L. Veltri, "Information centric networking over SDN and OpenFlow: Architectural aspects and experiments on the OFELIA testbed," *Computer Networks*, vol. 57,(2013), pp. 3207-3221.

[84] N. B. Melazzi, A. Detti, G. Mazza, G. Morabito, S. Salsano, and L. Veltri, "An openflow-based testbed for information centric networking", *Future Network & Mobile Summit (FutureNetw)*, (2012), pp.1-9.

[85] N. B. Melazzi, A. Detti, M. Pomposini, and S. Salsano, "Route discovery and caching: a way to improve the scalability of Information-Centric Networking", *IEEE Global Communications Conference (GLOBECOM)*, (2012), pp. 2701-2707.

[86] A. Ooka, S. Ata, T. Koide, H. Shimonishi, and M. Murata, "OpenFlow-based content-centric networking architecture and router implementation", *Future Network and Mobile Summit (FutureNetworkSummit)*, (2013), pp. 1-10.

[87] I. Carvalho, F. Faria, E. Cerqueira, and A. Abelem, "ContentFlow: An Introductory Routing Proposal for Content Centric Networks using Openflow", *API 7th Think-Tank Meeting*, (2012), pp. 1-2.

[88] X. N. Nguyen, D. Saucez, and T. Turletti, "Efficient caching in content-centric networks using OpenFlow", *INFOCOM 2013 Student Workshop*, (2013).

[89] J. Torres, L. Ferraz, and O. Duarte, "Controller-based routing scheme for Named Data Network", *Technical report, Electrical Engineering Program*, (2012).

[90] J. Ren, K. Pentikousis, C. Westphal, W. Liu, and J. Wang, "The Role of Virtualization in Information-centric Network Deployment", *E-LETTER*.

[91] S. Civanlar, M. Parlakisik, A. M. Tekalp, B. Gorkemli, B. Kaytaz, and E. Onem, "A qos-enabled openflow environment for scalable video streaming", *IEEE GLOBECOM Workshops (GC Wkshps)*, (2010), pp. 351-356.

[92] H. E. Egilmez, S. T. Dane, K. T. Bagci, and A. M. Tekalp, "OpenQoS: An OpenFlow controller design for multimedia delivery with end-to-end Quality of Service over Software-Defined Networks," *Signal & Information Processing Association Annual Summit and Conference (APSIPA ASC)*, (2012), pp. 1-8.

[93] R. Braden, D. Clark, and S. Shenker, "Integrated services in the internet architecture: an overview", RFC 1633, (1994), available online: http://tools.ietf.org/html/rfc1633.html, last visit:18.10.2014.

[94] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss, "An architecture for differentiated services", RFC 2475,(1998), available online: https://tools.ietf.org/html/rfc2475, last visit:18.10.2014.

[95] Network Configuration (NetConf), http://datatracker.ietf.org/wg/netconf, last visit:18.10.2014.

[96] Open Networking Foundation, "OpenFlow configuration and management protocol 1.0 (OF-Cinfig)", (2011), available online: https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow-config/of-config1dot0-final.pdf, last visit:18.10.2014.

[97] W. Kim, P. Sharma, J. Lee, S. Banerjee, J. Tourrilhes, S.-J. Lee, *et al.*, "Automated and scalable qos control for network convergence", *INM/WREN*, vol. 10, (2010), pp. 1-1, 2010.

[98] I. Owens and A. Durresi, "Video over Software-Defined Networking (VSDN)", *2013 16th International Conference on Network-Based Information Systems (NBiS)*, (2013), pp. 44-51.

[99] H. Kim and N. Feamster, "Improving network management with software defined networking," *Communications Magazine, IEEE*, Vol.51, No.2, (2013), pp.114-119, available online: http://dx.doi.org/ 10.1109/MCOM.2013.6461195, last visit:18.10.2014.

[100] P. Sharma, S. Banerjee, S. Tandel, R. Aguiar, R. Amorim, and D. Pinheiro, "Enhancing network management frameworks with SDN-like control," *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, (2013), pp.688-691, available online: http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=6573054&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D6573054, last visit:18.10.2014.

[101] S. Dotcenko, A. Vladyko, and I. Letenko, "A fuzzy logic-based information security management for software-defined networks," *Advanced Communication Technology (ICACT), 2014 16th International Conference on*, (2014), pp. 167-171, available online: http://dx.doi.org/10.1109/ICACT.2014.6778942, last visit:18.10.2014.

[102] H. Kim, A. Voellmy, S. Burnett, N. Feamster, and R. Clark, "Lithium: Event-driven network control," *Georgia Institute of Technology*, (2012), available online: smartech.gatech.edu/jspui/bitstream/1853/43377/1/GT-CS-12-03.pdf, last visit:18.10.2014.

[103] *Welcome to project BISmark*. available online: http://projectbismark.net, last visit:18.10.2014.

[104] S. Song, S. Hong, X. Guan, B.-Y. Choi, and C. Choi, "NEOD: network embedded on-line disaster management framework for software defined networking," *Integrated Network Management (IM 2013), 2013 IFIP/IEEE International Symposium on*, (2013), pp.492-498, available online: http://dx.doi.org/10.1109/ICACT.2014.6778942, last visit: 18.10.2014.

[105] J. Matias, B. Tornero, A. Mendiola, E. Jacob, and N. Toledo, "Implementing Layer 2 Network Virtualization using OpenFlow: Challenges and Solutions", *Proceedings of European Workshop on Software Defined Networking (EWSDN)*, (2012), pp:30-35, available online: http://dx.doi.org/10.1109/EWSDN.2012.18, last visit:17.10.2014.

[106] R. Nejbati, S. Azodolmolky, and D. Simeonidou, "Role of Network Virtualization in Future Internet Innovation", *Proceedings of 17th European Conference on Networks and Optical Communications (NOC)*, (2012), pp:1-4, available online: http://dx.doi.org/10.1109/NOC.2012.6249915, last visit:17.10.2014.

[107] I. M. Moraes, D. M. Mattos, L. H. G. Ferraz, M. E. M. Campista, M. G. Rubinstein, L. H. M. Costa, *et al.*, "FITS: A Flexible Virtual Network Testbed Architecture", *Computer Networks*, Vol.63, (2014), pp:221–237, available online: http://dx.doi.org/10.1145/1477942.1477944, last visit:17.10.2014.

[108] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, *et al.*, "Xen and the Art of Virtualization", *Proceedings of the nineteenth ACM symposium on Operating systems principles*, Vol.37, (2003), pp:164-177, available online: http://dx.doi.org/10.1145/1165389.945462, last visit:17.10.2014.

[109] J. Hurwitz, A. Nugent, F. Halper, and M. Kaufman, *Big Data for Dummies*, John Wiley & Sons, Inc., (2013), pp:7-35.

[110] C. Baker, A. Anjum, R. Hill, N. Bessis, and S. L. Kiani, "Improving Cloud Datacenter Scalability, Agility and Performance using OpenFlow", *Proceedings of 4th IEEE International Conference on Intelligent Networking and Collaborative Systems (INCoS)*, (2012), available online: http://dx.doi.org/10.1109/iNCoS.2012.118, last visit:17.10.2014.

[111] Oracle SDN (Software Defined Network), (2013), Available online: http://www.oracle.com/us/products/networking/virtual-networking/sdn/overview/index.html, last visit:17.10.2014.

[112] R. Raghavendra, J. Lobo, and K.-W. Lee, "Dynamic Graph Query Primitives for SDN-based Cloud Network Management", *Proceedings of the first workshop on Hot topics in software defined networks*, (2012), pp.97-102, available online: http://dx.doi.org/ 10.1145/2342441.2342461, last visit:17.10.2014.

[113] M. Al-Fares, S. Radhakrishnan, B. Raghavan, N. Huang, and A. Vahdat, "Hedera: Dynamic Flow Scheduling for Data Center Networks", *Proceedings of 7th USENIX conference on Networked systems design and implementation*, (2010), available online: http://dl.acm.org/citation.cfm?id=1855730, last visit:17.10.2014.

[114] H. Yang, J. Zhang, Y. Zhao, H. Li, S. Huang, Y. Ji, *et al.*, "Cross Stratum Resilience for OpenFlow-enabled Data Center Interconnection with Flexi-Grid Optical Networks", *Optical Switching and Networking,* Vol.11, (2014), pp:72-82, available online: http://dx.doi.org/10.1016/j.osn.2013.10.001, last visit:17.10.2014.

[115] J. Suh, H. Choi, W. Yoon, T. You, T. Kwon, and Y. Choi, "Implementation of a Content-oriented Networking Architecture (CONA): A Focus on DDoS Countermeasure", *Proceedings of European NetFPGA developers workshop*, (2010).

[116] C. YuHunag, T. MinChi, C. YaoTing, C. YuChieh, and C. YanRen, "A Novel Design for Future on-demand Service and Security", *Proceedings of 12th IEEE International Conference on Communication Technology*, (2010), pp:385-388, available online: http://dx.doi.org/10.1109/ICCT.2010.5689156, last visit:17.10.2014.

[117] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow Random Host Mutation: Transparent Moving Target Defense using Software Defined Networking", *Proceedings of the first workshop on Hot topics in software defined networks*, (2012), pp:127-132.

[118] X. Liu, H. Xue, X. Feng, and Y. Dai, "Design of the Multi-level Security Network Switch System Which Restricts Covert Channel", *Proceedings of 3rd IEEE International Conference on communication software and networks (ICCSN)*, (2011), 2011, pp:233-237, available online: http://dx.doi.org/10.1109/ICCSN.2011.6013582, last visit:17.10.2014.

[119] Y. Juba, H.-H. Huang, and K. Kawagoe, "Dynamic Isolation of Network Devices Using OpenFlow for Keeping LAN Secure from Intra-LAN Attack", *Procedia computer science,* vol. 22, pp: 810-819, (2013), available online: http://dx.doi.org/ 10.1016/j.procs.2013.09.163, last visit: 17.10.2014.

[120] K. Giotis, C. Argyropoulos, G. Androulidakis, D. Kalogeras, and V. Maglaris, "Combining OpenFlow and sFlow for an Effective and Scalable Anomaly Detection and Mitigation Mechanism on SDN Environments", *Computer Networks,* (2013), available online: http://dx.doi.org/ 10.1016/j.bjp.2013.10.014, last visit:17.10.2014.

[121] P. Phaal, "sFlow Specification Version 5", (2004), available online: http://sflow.org/sflow_version_5.txt, last visit: 17.10.2014.

[122] H. T. T. Luo, and T. Quek, "Sensor openflow: Enabling software defined wireless sensor networks", *IEEE Communications Letters,* Vol.16, No. 11, (2012), pp. 1896-1899, available online: http://dx.doi.org/10.1109/LCOMM.2012.092812.121712, last visit: 18.10.2014.

[123] A. K. P. Dely, and N. Bayer, "Openflow for wireless mesh networks*", Proceedings of the Computer Communications and Networks*, (2011), pp. 1-6, available online: http://dx.doi.org/10.1109/ICCCN.2011.6006100, last visit: 18.10.2014.

[124] Y. S. P. Baskett, W. Zeng, and B. Guttersohn, "SDNAN: Software Defined Networking in Ad hoc Networks of Smartphones", *Proceedings of* the Consumer Communications and Networking Conference, (2013), pp. 861-862, http://dx.doi.org/10.1109/CCNC.2013.6488568, last visit: 18.10.2014.

[125] Open Networking Foundation, "Wireless & Mobile", available online: https://www.opennetworking.org/images/stories/downloads/working-groups/charter-wireless-mobile.pdf, last visit:18.10.2014.

[126] *Security Challenges in SDN (Software-defined Networks)*. available online: http://www.sdncentral.com/security-challenges-sdn-software-defined-networks/, last visit:18.10.2014.

[127] M. Caesar, D. Caldwell, N. Feamster, J. Rexford, A. Shaikh, and J. van der Merwe, "Design and implementation of a routing control platform," *Proceedings of the 2nd conference on Symposium on Networked Systems Design & Implementation,* Vol.2, (2005), pp.15-28, available online: https://www.usenix.org/legacy/events/nsdi05/tech/full_papers/caesar/caesar_html/, last visit:18.10.2014.

[128] R. Oshana and S. Addepalli, "Networking Trends-Software Defined Networking, Network Virtualization and Cloud Orchestration," *Asia Power Arch. Conf,* (2012), available online: https://www.power.org/wp-content/uploads/2012/10/13.-FSL-SDN-Openflow-and-Cloud-computing-UPD_Rob-Oshana.pdf, last visit:18.10.2014.