Implementing SDN into Computer Network Lessons

Filip Holik, Josef Horalek
University of Pardubice,
Faculty of Electrical Engineering and Informatics,
Pardubice, Czech Republic.
filip.holik@student.upce.cz

***Abstract*—This paper describes the issue of introducing SDN to students of computer networks. The most important theoretical knowledge is summarized in the form of key points, students should know about. Practical experience is presented in the area of deployment of SDN in data centers with aim on connecting the existing knowledge from traditional computer networks. This connection is explained on problems of traditional networks in data centers and mitigation of these problems by using SDN. Learned information is then extended by presenting a practical demo application in Mininet environment. The application shows possible usage of SDN for making a data center more power-efficient. This application is put in context by the theory of power consumption of data center devices which can be significantly reduced if SDN are used. Purpose of the application is to motivate students to continue in research of SDN area.**

***Index Terms*—SDN; Network; Power Consumption; Data Centers.**

I. INTRODUCTION

Software Defined Networking (SDN) is a concept of separation of a network's control from the network's data plane. Undergraduate students of computer networks typically do not have any knowledge of this topic. Even postgraduate students' knowledge would be very limited and only theoretical mainly due to SDN's complexity. SDN requires experience from computer networks as well as programming. For this reason, the topic is not included in current Cisco Networking Academy courses [1] or in similar programs.

The most active topics of research on SDN in data centers nowadays, are performance issues often using optical links [2],[3], QoS [4],[5], and power efficiency [6],[7]. However, there is currently no research done on implementation of SDN into education.

Introducing SDN is nevertheless a very demanding task and the topic can be very easily misunderstood. For these reasons, this paper provides an approach to explain a real world example of implementation of SDN in a data center. A data center is a modern topic with which students have knowledge. They can thus easily design a data center's topology and write required configurations for building such a network. This example would help students in understanding the topic and to connect their theoretical knowledge with practical experiences of SDN. This method of education supports applicability of modern approaches of teaching like flipped teaching, whole brain teaching, or gamification [8].

II. SDN ESSENTIALS

SDN evolved in response to insufficient features of classical data networks. Complex and vendor specific configuration, proprietary software, lack of new features, or complicated management are just a few main examples of insufficiency found in these networks. SDN aim to eliminate these flaws by introducing two main concepts. Separation of data and control plane, allowing transparent expansibility of new features and eliminating need for proprietary software; as well as centralized control of the whole network allowing simplified configuration and management [9,10].

A. Components of SDN

First thing when introducing SDN to students is explanation of basic components. The emphasis is put only on the most important concepts. Whenever possible, the topic should connect with existing knowledge from computer networks.

i. SDN Devices

SDN devices have their own data plane, but management or control plane is left for the controller. In order to support SDN, a device has to have a flow table. A flow table consists of flow entries, whose structure, described in the current Open Flow specifications, contains: match fields, priority, counters, instructions, timeouts, cookies and flags [11]. Implementation of the flow table can be done in software using various data structures, or more commonly in hardware, using existing hardware-based table as CAMs (Content Addressable Memory) or TCAMs (Ternary CAM). Many vendors, combine these approaches and offer a limited number of hardware tables, which can be extended by usage of software tables. An example of this solution can be switch HP 3800, which offers one HW table and multiple SW tables.

ii. Controller

The controller is responsible for management and control of all SDN devices. This is done via Southbound API which is typically represented by a standardized OpenFlow protocol. There are also other protocols like NETCONF [12], which aims at interacting with a device's configuration. The controller also communicates with user applications via Northbound API which can be represented by various interfaces and is not therefore standardized like Southbound API. The most common Northbound APIs are: REST, Python, or Java. The most common controllers nowadays are OpenDaylight, ONOS, Ryu, and Floodlight. These

controllers typically contain a ranging variety of modules, ready to be implemented into the network.

iii. Applications

Applications contain the whole intelligence of the SDN. Logic programmed in the application is communicated via a controller with SDN devices. Theoretically, applications could communicate with switches directly, but this would be too dangerous - errors and mistakes could bring the switch down. Standardized API is thus a necessity.

It is necessary to mention to students, that without applications, SDN network will do nothing as switches will be dropping all the packets. Even the most basic networking functions have to be programmed. Fortunately, most of these functions are already included with the most common controllers. For example, controller POX include L2 learning switch, L3 learning switch, hub, load balancer, simple firewall, and many others [13].

B. Basic Types of SDN

SDN is a dynamic and modern topic and for these reasons, it can have many definitions. In general, three main types of SDN are being used [14].

i. Open SDN

This is the traditional view of SDN as separation of a network control layer from a forwarding layer. Open SDN is using OpenFlow protocol to manage individual devices and it is most widely supported by research communities and data center operators. Google has used this type of SDN on its backbone network (B4) since 2011. B4 is a private WAN network connecting Google's data centers. As a result of implementing centralized traffic engineering (TE) via SDN, average long-term links' utilization increased from 30% to 70% while many links can be safely utilized almost at 100%. This efficiency is made possible without risk of service unavailability [15].

ii. Hypervisor-Based overlay Network

An alternative type of SDN is called Hypervisor SDN. This type utilizes network virtualization where a physical network is overlaid by a virtual network. This solution mitigates the need for VLANs by tunneling all data traffic. Also changes in virtual networks can be performed in a minimum time. On the other hand, overlay SDN is not dealing with physical issues like setting of QoS or modification of physical devices. Example of Overlay Networks are SDN VE (SDN for Virtual Environment) by IBM [16] and VMware NSX [17].

iii. SDN Over API

The third type uses existing traditional network functions which can be remotely controlled via legacy API protocols like SNMP, CLI, or via a modern RESTful API, which uses HTTP(S).

An example of SDN over API is Cisco onePK, which is using a Cisco ONE controller and allows SDN functionality to run even on legacy devices [18].

iv. Modern Approaches to SDN

Modern approaches to SDN are either trying to connect reliable functionality of traditional network with dynamic features of SDN; or they are trying to implement new functionality, which would make implementation of SDN easier. There are currently multiple paths in the research community, some of them implemented in various levels of usability. Purpose of this section is to show students modern trends in SDN and its possible usage in coherence with traditional networking.

Intent Networking. A traditional imperative way expresses configuration via standard commands like "ip route 0.0.0.0 0.0.0.0 Gi0/0". Intent networking is, instead, expressing configuration in a declarative way. That means expressing "what should be done", not "how to achieve it". An example would be "Create a route from A to B with minimum latency". The main advantage of this approach is that intents are not vendor or topology specific and can be still valid if the network will change. One of the implementations of Intent Networking called Group Based Policy is included in OpenDaylight Platform [19].

I2RS. The Interface to the Routing System (I2RS) - also called Hybrid SDN - is interacting with a Routing Information Base (RIB), instead of a Forwarding Information Base (FIB) as OpenFlow does. In this way, SDN functionality can be added to the existing routing decisions. The approach is therefore mixing an existing configuration with additional features like optimized routing, rapid re-routing, or collecting topology information [20].

OPLFLEX. An Open Policy Protocol (OPFLEX), originally developed by Cisco, uses declarative control like intent networking. In this concept, networking devices are represented as objects, which promise to reach and retain a state without specifying how to do it. OPFLEX also leaves intelligence on network devices, but allows definition and enforcement of various policies. Bidirectional communication also allows gathering information like events, statistics, and fault information. OPFLEX is also implemented in OpenDaylight [19].

v. Virtualization of SDN

Students of Cisco Networking Academy have knowledge of using network simulators like Cisco Packet Tracer [1] or emulators like GNS3 [21]. For SDN emulation, a different environment has to be used. Currently, there are these options of virtualization:

a) Mininet: an Open Source network emulator which can run virtual hosts, switches, and controllers inside one virtual machine or deployed on real hardware [22].
b) Cisco VIRL (Virtual Internet Routing Lab): a network simulation platform allowing virtual machines to run with Cisco's network operating systems like IOS, IOS XE / XR, NX-OS, and ASA. Currently, Cisco is offering an academic version of VIRL, which can run up to 20 nodes and can be purchased for $79.99 per year [23].
c) Virtualized platform: own virtualized network can be built in OpenStack using software switches like Open vSwitch or Cisco Nexus 1000v [24]. A particular controller can be installed on top of the virtual machine.

All of these solutions are using virtualization and thus have their advantages and disadvantages, if compared to physical deployment. For academic purposes, features of virtualization like low price and easy reconfiguration

outweigh the main advantage of physical hardware - which is performance.

vi. Mininet Essentials

For academic purposes, the easiest approach allowing most of the features, is to use the Mininet. It can be easily distributed and deployed due to its packaging in a VM. It also supports multiple controllers and additional ones can be easily added. Also it does not have excessive hardware requirements. For these reasons we choose it as a main tool for our integration of SDN into education. Students should be familiar with Mininet, at least on the basic level of the official walkthrough [25].

III. SDN DATA CENTERS

A brief theory of comparing classical data networks and SDN in a data center field is presented. The last part of this section describes the current state of the power consumption of typical data centers and its predicted growth in the following years. The purpose of this section is to support students' knowledge about the topic of our example application.

A. Classical Data Networks Versus SDN

Classical networks in a modern data center have several issues [14]. The first is fixed MAC address table size which is implemented in hardware and its size is determined by the vendor of the device. Even while there are special devices intended for data centers, this table can get full, if there is a massive number of virtual devices each having their own MAC address. Once the MAC address table gets full, the switch has to flood all incoming packets to all ports except the receiving one. This flood will result in significant decrease of a network's performance. Another issue is in 802.1Q protocol, which supports maximally 4096 VLANs (Virtual Local Area Networks). In data centers containing thousands of users, this number can be limiting. One solution is to use MPLS (Multiprotocol Label Switching) which does not have this limitation. Both of these problems can be solved by usage of hypervisor-based SDN which uses MAC addressing only on a virtual tunnel's end points. Tunneling technologies can also address millions of networks; solving the maximum VLAN issue. MPLS can also be easily replaced by SDN [26].

Other issues are connected with the STP (Spanning Tree Protocol). This protocol prevents creation of link loops on the second layer of the ISO/OSI model. By default, a STP blocks duplicated links to prevent these loops, which makes these links unusable. This state is not ideal, especially in data centers. It is possible to configure a STP to use these duplicated links by setting up separate VLANs, but configuration is not transparent nor dynamic. Another STP issue is relatively slow convergence if network change occurs. This is especially true if a STP is compared to tuned layer 3 protocols like OSPF or EIGRP. A STP can be replaced by more appropriate and modern L2 protocols like 802.1aq, or Transparent Interconnection of Lots of Links (TRILL). Another approach is to use L3 protocols like OSPF, IS-IS or EIGRP which can load balance and thus use redundant links. Also their convergence time can be relatively fast. Unfortunately even usage of these protocols does not ensure optimal functionality. As current research shows [26], SDN implementation can achieve much faster and efficient functionality. Moreover by replacing traditional networks by SDN, mentioned problems can be also solved.

B. Other Issues in Data center

As data centers are becoming complicated [27], issues with effective management, failure recovery and multiple tenancy are rising.

Effective management is needed for flexible adding, relocating, and removing of components. These changes are becoming more rapid with the deployment of virtual servers. A traditional approach via manual reconfiguration with changes of physical infrastructure is slow and can bring errors into the network. If SDN technology is used, virtual tunnels, Open SDN, or SDN over API can be used for manipulation with configuration of physical devices. Automated tasks reacting on network changes can be set up as well.

Failure recovery in traditional data centers can be non-deterministic, so time needed to complete recovery is impossible to predict. Furthermore, with increasing size of data center, this time will increase. If SDN is used, the controller has an overview of the whole network (knows all the routing rules, sees complete topology, and can communicate with other sources of information), which makes recovery predictable.

Data centers hosting multiple users have to deal with multiple tenancy. Each user has to be isolated and separated to guarantee proper functionality and security. In traditional networks, this can be achieved with VLANs, but their limitations were mentioned. On the other hand, an SDN can provide separation by network virtualization, or via MAC-in-MAC or Q-in-Q encapsulation.

C. Data Center Power Consumption

Global data centers in 2010, were using up to 1.5% of total electricity used worldwide and up to 2.2% in the U.S. [28]. In 2013, data centers located in the U.S. consumed approximately 91 billion kWh and it is estimated that this number will increase to roughly 140 billion kWh by 2020. Even more important is the fact, that around 50% of this energy is wasted due low server utilization [29]. The same study shows, typical servers' power efficiency drops significantly as utilization decreases. A typical server utilized at 10% still uses 30% – 60% of its maximum power consumption.

Additional power consumption is caused by the networking devices in a data center. To ensure reliable, flexible and fast communication, network devices and links are typically built redundantly, which cause significant additional power consumption. The most common networking device in data centers is a switch, which can have hundreds to thousands of ports. Unlike in servers, power consumption of a switch is mostly dependent on its ports and their speeds. One solution to save power from ports, is a power scaling algorithm, which can slow a port down if it is not fully utilized [6]. As the authors state, a typical 1 Gbps port consumes 1080mW, but if switched to 100Mbps, it consumes only 112mW; and on 10Mbps it consumes 52mW.

IV. Example Application of SDN in Data Centers

This section explains functionality of our example application which is presented to students as their first practical scenario of SDN.

A. SDN Application Introduction

Our proposed application example is aiming at decreasing power consumption of a typical data center with thousands of virtual servers. These servers can be distributed among tens to hundreds of physical servers which are connected via networking infrastructure. The multiple servers are hosting the same users and their application for greater performance and increased availability in the case of failure. Demands for these services are varying; typically it is much lower during night hours. For these reasons it makes sense to shut down the servers if they are not needed. In this case it is also necessary to modify networking infrastructure to correspond to new topology. Unused networking devices can be turned off, which can also contribute to lowering of the total data center's power consumption. Unlike servers, networking devices does not typically have so many options in scaling performance and power consumption based on actual devices' utilization.

Our SDN application simulates this change in topology with redirecting of data flow to different paths along with shutting down unused networking devices. Application is written in Python for a POX Controller. We are using a Mininet environment for the testing. In our scenario, only the networking part is being tested and evaluated. In a real world data center, application would have to be modified to communicate with other applications which would monitor servers' utilization and their shutdown/turn on. This integration is not possible in the Mininet environment due to server application's close dependency on a specific hardware.

For simplicity, we are using minimal topology displayed in Figure 1. Topology contains one host (h1) accessing one of the servers (h2, h3) via data centers' infrastructure (switches s1, s2, s3) which are controlled by the POX controller.

B. Functionality of the Application

The application itself is monitoring utilization of network devices and in the case of low utilization it simulates shutting down part of the network and redirecting data flow to another part. In a real data center, this monitoring would probably be moved to another application for monitoring hardware utilization of servers instead of network devices. That application would then communicate with our application and call its functions.

Firstly, the administrator has to specify maximum utilization of the link in packets per minute. This number is then used in computing total utilization.

```
Utilization = (f.packet_count / MAX_UTIL) * 100
```

Application contains timer set to 60 seconds in which switches are periodically probed and their statistics are collected.

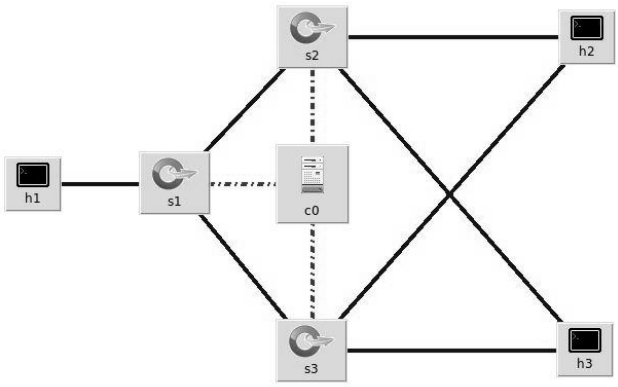

Figure 1: Testing Mininet topology

```
Timer(60, _minuteTimer, recurring=True)
```

Number of packets received in one minute is compared to set thresholds. If it is lower and the network is operating at full performance, part of the network can be shut down. On the other hand, if the number is higher than the set threshold and part of the network is down, it can be enabled.

```
if utilization < 50:
          if(powerSafeEnabled == 0):
             enablePowerSafeMode()
          else: log.info("Already in
power safe…")
else:
          if(powerSafeEnabled == 1):
             disablePowerSafeMode()
          else: log.info("Already in
full performance…")
```

After comparing set values and performing selected actions, controller sends OFPFC_DELETE message to clear all switches' statistics and timer will start again.

C. Application Summary

This testing example introduces possible usage of SDN in a data center. Application can be used to lower total power consumption of data centers' network infrastructure. In extension with an additional application controlling servers' hardware, power consumption of the data center can be effectively managed.

V. Conclusion

SDN is a modern topic which every student of computer networks should know about. We proposed an approach to introduce SDN technology in a form of seminar or extended lecture. Knowledge of typical students of current computer network courses like Cisco Networking Academy are didactically used in our approach. In order to show practical usage of SDN, we proposed an example demo application which is using topology from a data center. Application is collecting network utilization and is simulating shutting down or enabling parts of the network in order to optimize the network's power consumption. This example is built in order to motivate students of computer networks to learn more about SDN.

ACKNOWLEDGMENT

This work and contribution is supported by the project of the student grant competition of the University of Pardubice, Faculty of Electrical Engineering and Informatics, Intelligent Smart Grid networks protection system, using software-defined networks, no. SGS_2016_016.

REFERENCES

[1] Cisco, (2016). "Cisco Networking Academy". Retrieved from https://www.netacad.com/, on 8 Jan 2016.

[2] Zhao, Y., Zhang, J., Yang, H. and Yu, X., "Data center optical networks (dcon) with openflow based software defined networking (SDN)". *In Communications and Networking in China (CHINACOM), 2013 8th International ICST Conference*, pp. 771-775, 2013.

[3] Tu, R., Wang, X., Zhao, J., Yang, Y., Shi, L. and Wolf, T.,. "Design of a load-balancing middlebox based on SDN for data centers". *In 2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS*), pp. 480-485, 2015.

[4] Miao, W., Agraz, F., Peng, S., Spadaro, S., Bernini, G., Perelló, J., Zervas, G., Nejabati, R., Ciulli, N., Simeonidou, D. and Dorren, H.,. "SDN-enabled OPS with QoS guarantee for reconfigurable virtual data center networks". *Journal of Optical Communications and Networking*, vol. 7, no. 7, pp.634-643, 2015.

[5] Wang, J.M., Wang, Y., Dai, X. and Bensaou, B., "SDN-based multi-class QoS-guaranteed inter-data center traffic management". *In Cloud Networking (CloudNet), 2014 IEEE 3rd International Conference*, pp. 401-406, 2014.

[6] Nam, T.M., Thanh, N.H., Thu, N.Q., Hieu, H.T. and Covaci, S., " Energy-aware routing based on power profile of devices in data center networks using SDN". *In Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2015 12th International Conference*, pp. 1-6, 2015.

[7] Han, Y., Li, J., Chung, J.Y., Yoo, J.H. and Hong, J.W.K., " SAVE: Energy-aware Virtual Data Center embedding and Traffic Engineering using SDN". *In Network Softwarization (NetSoft), 1st IEEE Conference*, pp. 1-9, 2015.

[8] Tretinjak M, Bednjanec A, Tretinjak M "Application of modern teaching techniques in the educational process". *In: MIPRO*, pp. 628-632, 2014.

[9] Santos, M.A., Nunes, B.A., Obraczka, K., Turletti, T., de Oliveira, B.T. and Margi, C.B., "Decentralizing SDN's control plane". *In 39th Annual IEEE Conference on Local Computer Networks*, pp. 402-405, 2014.

[10] Feamster N, Rexford J, Zegura E., (2014). " The Road to SDN: An Intellectual History of Programmable Networks". Retrieved from: http://goo.gl/tVA5vy, on 8 Jan 2016.

[11] The ONF (2015). "OpenFlow Switch Specification Version 1.5.1". Retrieved from: https://goo.gl/gzPDhS, on 8 Jan 2016.

[12] Enns E et al (2011). "Network Configuration Protocol (NETCONF), RFC 6241". Retrieved from: https://tools.ietf.org/html/rfc6241 , on 8 Jan 2016.

[13] NOXRepo. (2015). "About POX". Retrieved from: http://www.noxrepo.org/pox/about-pox/, on 24 September 2015.

[14] Goransson P, Black C. "Software defined networks: a comprehensive approach". *Amsterdam: Morgan Kaufmann*, pp. 325, 2014. ISBN 978-0-12-416675-2.

[15] Sushant J et al (2013). " B4: Experience with a Globally-Deployed Software Defined WAN. In: SIGCOMM, China". Retrieved from: http://goo.gl/7XLKgD, on 5 November 2015.

[16] IBM (2015). "IBM Software Defined Networking". Retrieved from: http://goo.gl/5IcHuV, on 9 November 2015.

[17] Hatem Naguib. (2013). "VMware Network Virtualization". Retrieved from: http://goo.gl/Fm87SN, on 5 November 2015.

[18] Cisco DevNet (2015) onePK. Retrieved from: https://developer.cisco.com/site/onepk/, on 5 November 2015.

[19] OpenDaylight Community (2015) OpenDayligh User Guide – Lithium. Retrieved from:http://goo.gl/8D7tAz, on 7 November 2015.

[20] Haas J, Hares S. (2016). " I2RS Ephemeral State Requirements", IETF Draft. Retrieved from: https://goo.gl/tge175 , on 10 March 2016.

[21] GNS3 Technologies (2015), "GNS3, The software that empowers network professionals". Retrieved from: http://www.gns3.com/, on 9 November 2015 .

[22] Mininet team (2015) Mininet. Retrieved from: http://mininet.org,on 9 November 2015.

[23] Cisco, (2015), "Introducing Virtual Internet Routing Lab". Retrieved from: http://virl.cisco.com/, on 9 November 2015.

[24] Graf T (2013) Underneath OpenStack Quantum: Software Defined Networking with Open vSwitch.Retrieved from: http://goo.gl/gSvHQC., on 12 November 2015.

[25] Mininet Team, (2015). "Mininet Walkthrough". Rtrieved from: http://goo.gl/kt9JW8, on 9 November 2015.

[26] Dorsch, N., Kurtz, F., Georg, H., Hägerling, C. and Wietfeld, C., "Software-defined networking for smart grid communications: Applications, challenges and advantages". *In Smart Grid Communications (SmartGridComm), 2014 IEEE International Conference*, pp. 422-427, 2014.

[27] Li, D., Shang, Y. and Chen, C., "Software defined green data center network with exclusive routing". *In IEEE INFOCOM 2014-IEEE Conference on Computer Communications*, pp. 1743-1751, 2014.

[28] Koomey J.G., (2011), "Growth in data center electricity use 2005 to 2010". Oakland, CA: Analytics Press. Retrieved from: http://goo.gl/sKEbwy, on 5 November 2015.

[29] NRDC - Natural Resources Defense Council (2014), "Data Center Efficiency Assessment". New York. Retrieved from: http://goo.gl/N1dKDG. on 5 November 2015.